Technische Universität Graz
Fakultät für Maschinenbau und Betriebsinformatik

Verbesserung der Ladeprognose von batteriebetriebenen Fahrzeugen durch maschinelles Lernen / Künstliche Intelligenz

Masterarbeit

Verfasser:
Raffael Sacher

Betreuer:
Dipl. Ing. Dr.techn. Nikolaus Furian
Betreuer Daimler AG:
Dr. Osman Sahin

Institut für Maschinenbau und Betriebsinformatik

Graz und Stuttgart, Februar 2019

Eidestattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtliche und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.
Das TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

.....
Raffael Sacher

Gleichheitsgrundsatz

Aus Gründen der Lesbarkeit wurde in dieser Arbeit darauf verzichtet, geschlechterspezifische Formulierungen zu verwenden. Es wird hiermit jedoch ausdrücklich festgehalten, dass die bei Personen verwendete maskuline Form für beide Geschlechter zu verstehen ist.

SPERRVERMERK

Die nachfolgende Masterarbeit mit dem Titel

Verbesserung der Ladeprognose eines batteriebetriebenen Fahrzeuges durch maschinelles Lernen / Künstliche Intelligenz

Autor: Raffael Sacher

Enthält vertrauliche Daten des Unternehmens Daimler AG. Veröffentlichungen oder Vervielfältigungen der Arbeit, auch nur auszugsweise, sind ohne ausdrückliche Genehmigung der Daimler AG nicht gestattet. Die Masterarbeit ist nur der Bibliothek der Technischen Universität Graz zugänglich zu machen.

Die Sperrung der Arbeit erstreckt sich über einen Zeitraum von 5 (fünf) Jahren

Kurzfassung

Batteriebetriebene Fahrzeuge reduzieren den Energieverbrauch und die Umweltverschmutzung im Vergleich zu konventionellen Fahrzeugen mit Verbrennungsmotor. Die limitierende Reichweite und die Ladezeit der Batterie erzeugen neue Herausforderungen. Die Genauigkeit der Vorhersage, wieviel Zeit noch benötigt wird um die Batterie zu laden, ist daher grundlegend wichtig und hilft dem Nutzer seine Reisepläne zu gestalten und senkt die Sorge bezüglich der Reichweite. In dieser Arbeit wird geprüft ob mittels maschinellem Lernen eine Vorhersage der Ladezeit möglich ist und ob ein datenbasiertes Modell bessere Ergebnisse liefert als ein physikalisches Modell. Zuerst wurden die Daten analysiert und auf Repräsentativität und Qualität geprüft. Es wurden verschiedene Ansätze verglichen, um die vorhandenen Daten in ein überwachtes maschinelles Lernproblem zu transformieren. Anschließend wurden zwei Modelle, Lineare Lasso Regression und Zufallswald (engl. Random Forest) ausgewählt und hinsichtlich der Genauigkeit verglichen. Um die Genauigkeit der Modelle zu verbessern wurde die Methode der Rastersuche (engl. Grid Search) verwendet. Diese dienten dazu, die besten Hyperparameter der Modelle zu finden. Anschließend wurden die Ergebnisse erläutert und diskutiert. Letztlich wurden Handlungsempfehlungen und Ergebnisse zusammengefasst und ein Ausblick gegeben.

Keywords: Elektromobilität, Batterie, Maschinelles Lernen, Künstliche Intelligenz

Abstract

Battery electric vehicles reduce the energy consumption and air pollution in comparison to conventional vehicles with an internal combustion engine. The limited range and the charging time of electric vehicles create new challenges. Hence, the accuracy of the prediction, how much time is needed to charge the Battery, is crucial and helps the driver to plan their travels and decreases the worries and concerns regarding the driving range.
During this work, the basic question, if machine learning can be used to predict the battery charging time is answered. Furthermore a comparison between the built machine learning model and the actual physical model is given.
First, the data was analyzed and checked regarding quality and representativeness. Three different approaches were compared, to transform time series into a machine learning problem. Consequently two different models, linear lasso regression and random forest regression, were implemented and compared in terms of prediction quality. Accordingly, a grid search method was used to improve the model performance by searching for the best hyperparameter set. Furthermore the results were presented and discussed. In conclusion a complete overview about the thesis and a recommendation for further actions were given.

Keywords: Electric Mobility, Battery, Machine Learning, Artificial Intelligence

Abkürzungsverzeichnis

Area under the curve	*AUC*
Artificial Intelligence	*AI*
Künstliche Intelligens	*KI*
Lineare Lasso Regression	*LLR*
Maschinelles Lernen	*ML*
Mean square error	*MAE*
Principal component analysis	*PCA*
Random Forest	*RF*
Root mean square error	*RMSE*
State of Charge	*SOC*

Inhaltsverzeichnis

1 Einleitung

Die Weiterentwicklung der Elektromobilität ist ein zukunftsweisendes Thema der deutschen Industrie. Mehr noch: Elektrofahrzeuge können ein wichtiger Baustein der Energiewende werden [1]

Alle deutschen Automobilhersteller starten in den nächsten Jahren in die Offensive der Elektromobilität. Kernthemen wie Reichweite, Ladeinfrastruktur und Batterieerzeugung bringen neue Herausforderungen mit sich, um in eine elektrische Zukunft zu starten. Die Reichweite und die Dauer des Wiederaufladens der Batterie, wird unter anderem über den Erfolg des Elektrofahrzeuges entscheiden.
Die Betankung eines Fahrzeuges mit Verbrennungsmotors ist in wenigen Minuten erledigt, das Laden eines Elektrofahrzeuges kann je nach Ladeleistung jedoch Stunden dauern. Umso wichtiger ist es für den Kunden, genau zu wissen, wieviel Zeit für eine Vollladung der Batterie benötigt wird. In dieser Arbeit wird untersucht, ob mittels maschinellem Lernen die benötigte Zeit für eine Vollladung der Batterie vorhersagbar ist und ob dieser Ansatz eine bessere Präzision liefert als ein Berechnungsmodell das auf physikalischen Größen basiert.

In Kapitel 2 wird grundlegend auf das Laden von Batterien eingegangen. Es wird der aktuelle Stand der Technik der aktuellen Ladestrategien eines Ladevorganges erklärt.
In Kapitel 3 wird grundlegend maschinelles Lernen erklärt und auf die verschiedenen Arten des maschinellen Lernens eingegangen. Anschließend wird das Vorgehensmodell von der Datenaufbereitung bis zur Modellbildung und Evaluation erklärt. Es werden die verschiedenen Möglichkeiten zur Datenaufbereitung und der Modellevaluation aufgezeigt.
In Kapitel 4 wird der Anwendungsfall der Prognose der Restladezeit der Batterie beschrieben und erläutert. Im Unterkapitel 4.1 wird der aktuelle Stand der Technik der Hybridfahrzeuge in Bezug auf die verbaute Ladetechnik des Projektpartners Daimler AG erläutert. Im Unterkapitel 4.2 wird auf die Datenerfassung, die Datengrundlage, den Dateiaufbau sowie die verwendeten Signale eingegangen.
In Unterkapitel 4.3 wird auf die Datenaufbereitung, sowie auf die verwendeten Ansätze um Zeitreihen zu transformieren eingegangen.
In Kapitel 4.4 wird die iterative Modellbildung chronologisch erklärt, beginnend mit der Fragestellung ob es grundsätzlich möglich ist, diesen Anwendungsfall mittels maschinellem Lernen abzubilden.
In Kapitel 5 werden die Ergebnisse interpretiert, diskutiert und ein Vergleich der Modelle erstellt.
In Kapitel 6 wird ein Zusammenfassung über die gesamte Arbeit inklusive der der wichtigsten Ergebnisse gegeben. Anschließend werden Handlungsempfehlungen und ein Ausblick für den weiteren Vorgang dieses Projektes gegeben.

2 Grundlagen, Stand der Technik Laden

Wie beim Laden von Handys und Laptops, ist auch beim Laden eines Fahrzeuges ein spezielles Ladegerät mit passender Ladestrategie notwendig. Hierbei wird elektrische Energie in chemische Energie umgewandelt und umgekehrt. Beeinflussende Kenngrößen hierbei sind der Ladestrom, die Ladespannung, die Kapazität der Batterie sowie die Ladezeit beziehungsweise die Laderate. Eine Laderate von 1C bedeutet, dass die Batterie mit einem Strom geladen wird, so dass es in einer Stunde zur Vollladung kommt. Bei einer Laderate von 2 C wird mit doppeltem Strom geladen und somit verkürzt sich die Zeit bis zur vollständigen Ladung auf eine halbe Stunde [3]. Im folgenden Kapitel wird der Stand der Technik in Bezug des Ladeprinzips, des Ladevorganges und der Ladeprognose vorgestellt. Dieser Abschnitt gibt einen Einblick in die Thematik und hilft ein grundlegendes Verständnis für die Themenstellung zu entwickeln.

2.1 Ladeprinzip

Das Laden eines batteriebetriebenen Fahrzeuges kann die Batterieperformance sowie die Lebenszyklen der Batterie stark beeinflussen. Ein Überladen der Batterie kann zu Batterieschäden führen, eine Tiefentladung hingegen kann zur Reduktion der Energiekapazität führen. Daher ist es wichtig eine passende Ladestrategie anzuwenden [5].
Der Ladezustand (engl. state-of-charge, Abkürzung SOC) ist einer der wichtigsten Parameter um den Zustand der Batterie zu beschreiben. Hierbei ist Q, die bei der Spannung U verfügbare elektrische Ladung (Coloumb = As) und Q_0 die Kapazität der vollgeladenen Batterie, wie in Gleichung 1 erläutert. Der SoC bewegt sich im Bereich von 0 und 100%, wobei 0 eine leere Batterie und 100% eine volle Batterie wiederspiegelt [4].

$$SoC = \frac{Q(U)}{Q_0} \qquad (1)$$

Für das Laden von Batterien gibt es mehrere Strategien. Die weit verbreitetste, in den meisten batteriebetriebenen Fahrzeugen verwendete Strategie wird nachfolgend vorgestellt. Das Laden einer Lithium Batterie basiert auf dem Prinzip des Ladens mit konstantem Strom und konstanter Spannung wie in Abb. 1: dargestellt. Das Laden ist unterteilt in drei Bereiche, Bereich 1 zeigt das Laden mit konstantem Strom (Constant-current charging) und Bereich 2 zeigt das Laden mit konstanter Spannung (Constant-Voltage charging). Im Bereich 3 wird die Spannung reduziert um die Batterie auf einem Ladezustand von 100% zu halten. Der Vorteil dieser Strategie ist, dass der Batteriestrom im Bereich 1 konstant gehalten wird und somit die anfängliche Stromspitze vermieden wird. Dies würde zu einer frühzeitigen Alterung der Batterie führen. Weiters wird das Risiko der Überladung im Bereich 2 durch den sinkenden Strom minimiert, was einen weiteren Vorteil mit sich bringt [4].

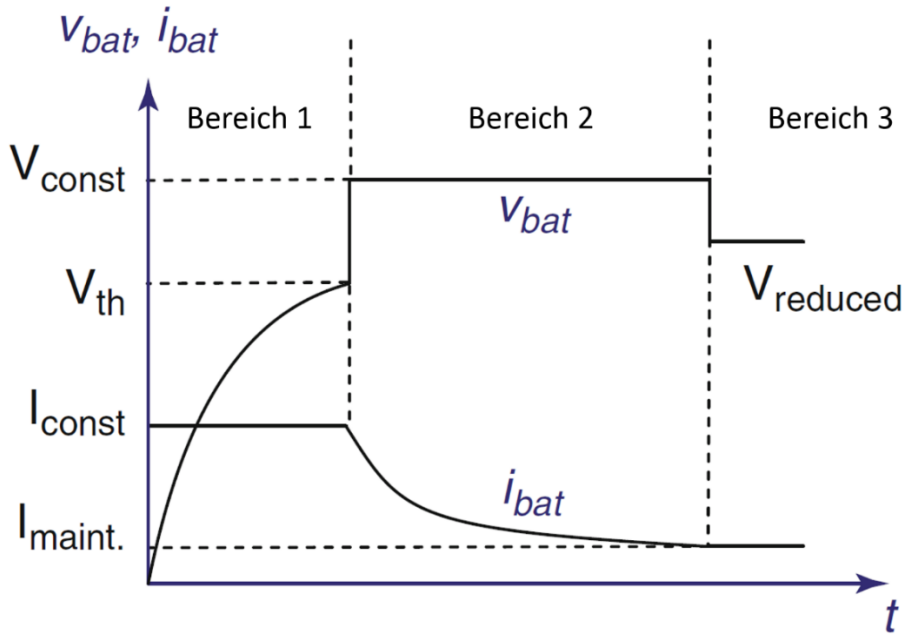

Abb. 1: Typische Ladekurve angelehnt an [5]

Zuerst wird die Batterie mit konstantem Strom geladen, bis ein voreingestellter Wert V_{const} erreicht ist. Danach wird die Batteriespannung konstant gehalten und der Strom verringert um die Batterie vollständig zu laden. Zuletzt wird die Batteriespannung reduziert um die Verluste der Selbstentladung der Batterie zu kompensieren und somit den Ladezustand zu halten.

2.2 Ladevorgang

Ein Ladevorgang in einem batteriebetriebenen Fahrzeug folgt dem Ladeprinzip, wie in Abschnitt 2.1 erklärt. Betrachtet man nun den ganzen Vorgang, von der Stromversorgung bis zu dem Energieinhalt der schlussendlich in der Batterie landet, wird ersichtlich, dass weitere Kenngrößen Einfluss haben. Das Prinzip des Energieflusses ist in Abb. 2: dargestellt.
Parameter wie Temperatur der Batterie, maximal abgebbare Leistung der Ladeinfrastruktur, Gleichstrom oder Wechselstrom, die maximale Leistung des On Bord Ladegerätes, die benötigte Leistung der Nebenverbraucher, die Verlustleistung der Nebenverbraucher, die gewünschte Abfahrtszeit des Kunden, Alter der Batterie, sowie die Anzahl an Ladezyklen spielen dabei eine Rolle.

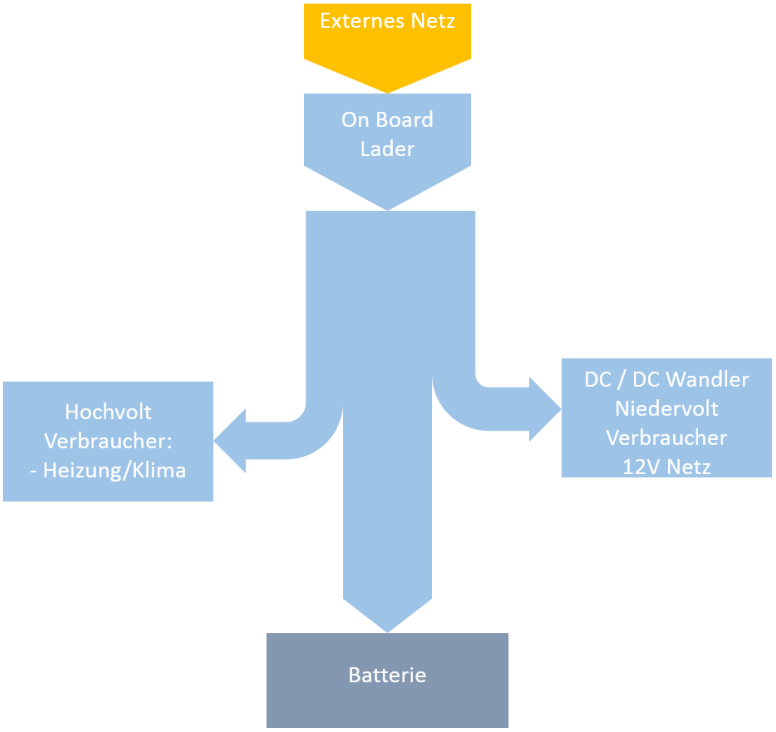

Abb. 2: Überblick Energiefluss Ladevorgang angelehnt an [3]

Die Energie fließt vom externen Stromnetz in das eingebaute Ladegerät und teilt sich danach auf in den Hochvolt Verbrauch, Niedervolt Verbrauch und in die Hochvolt Batterie.
Zu den „Hochvolt – Verbrauchern“ zählt man die Klimaanlage sowie die Heizung. Der Niedervolt Verbrauch wird von einem Gleichstrom Wandler auf die Bordspannung von 12V gewandelt und stellt die Versorgung der verbauten Steuergeräte, der Kühlpumpen und der anderen Kleinverbrauchern sicher. Somit wird die restliche, von den Verbrauchern nicht benötigte Energie, in die Batterie geladen. Ist die benötigte Leistung der Nebenverbraucher größer, als die vom Netz bereitgestellte Leistung, kann es zur Entladung der Batterie kommen. Hierbei dreht sich der Energiefluss um und es wird Energie aus der Batterie entnommen. Dies kann unter anderem dann vorkommen, wenn die Standheizung oder die Vorklimatisierung des Fahrzeuges während des Ladevorganges eingeschaltet ist. Hierbei wird das Fahrzeug während des Ladevorganges auf eine gewünschte Temperatur erwärmt oder gekühlt, um bei Fahrtantritt dem Nutzer ein vorgewärmtes bzw. vorklimatisiertes Fahrzeug bereit zu stellen. Je nach Temperaturunterschied des Fahrzeuges zur Außentemperatur, variiert die dazu benötigte Leistung. Beispielsweise wird bei einer Außentemperatur von -20°C, das Aufheizen des Fahrzeuges auf eine Raumtemperatur von +20°C einen beträchtlichen Energieaufwand ausmachen. Wird dieser Leistungsbedarf vom externen Netz nicht gedeckt, wird diese Leistung von der Batterie kompensiert. Somit ist der Ladevorgang aktiv, es wird jedoch Energie aus der Batterie entnommen.

2.3 Ladeprognose

Die benötigte Zeit um eine Batterie vollzuladen wird als Ladezeit oder Restladezeit bezeichnet. Ein konstanter Kritikpunkt der Elektromobilität ist die Restladezeit der Batterie, da es einen bedeutenden Unterschied macht, ob der Nutzer bei einer Weiterfahrt eine ganze Nacht, eine Stunde oder nur wenige Minuten warten muss. Die Prognose, wieviel Zeit benötigt wird um eine Batterie zu laden, ist somit essentiell und für die Kundenzufriedenheit sehr wichtig.
Diese Prognose kann aufgrund **physikalischer Berechnungen** erfolgen, indem ein Batteriekennfeld je nach Ladezustand der Batterie und Temperatur herangezogen wird. Somit kann die maximale Leistung der Batterie zu diesem Ladezustand und der jeweiligen Temperatur bestimmt werden. Mittels der maximalen verfügbaren Leistung der Infrastruktur, kann somit die verfügbare Ladeleistung bestimmt werden. Mittels der verfügbaren Ladeleistung der Infrastruktur, der maximal möglichen Leistung der Batterie und dem aktuellen Ladezustand der Batterie kann die Restladezeit der Batterie berechnet werden.
Ein neuer Ansatz diese Prognose zu bestimmen, ist ein **datenbasiertes Modell**. In [3] wird eine Studie vorgestellt, in der Lade- und Entladedaten von 70 Elektrofahrzeugen in China ausgewertet wurden. Diese Daten wurden verwendet um ein Regressionsmodell zu bilden und eine Vorhersage über die Ladezeit der Batterie zu prognostizieren. Hierbei bildete eine Anzahl von 41.400 Ladevorgängen die Datengrundlage.

3 Grundlagen maschinelles Lernen

3.1 Definition

Laut Elaine Rich ist Künstliche Intelligenz (engl. Artificial Intelligence) wie folgt definiert:

> *Artificial Intelligence is the study of how to make computers do things at which, at the moment, people are better [35].*

In vielen Bereichen der Wissenschaft, Technik, Biologie, Medizin etc. wird versucht, Phänomene basierend auf vergangenen Beobachtungen oder Messungen vorherzusagen. Zum Beispiel, wird von Meteorologen versucht das Wetter anhand der Klimabedingungen der Vortage vorherzusagen. Hierbei wird versucht, die Zielvariable, in diesem Fall das Wetter, basierend auf beobachteten Variablen vorherzusagen. Um diese Zielvariable präzise vorherzusagen, ist oftmals eine hohe Anzahl an zu beobachtenden Variablen nötig. Die Möglichkeit, dies von Experten mittels Gleichungssystemen zu bestimmen, wird in immer komplexer werdenden Systemen häufig überschritten. Hierbei hat sich Künstliche Intelligenz beziehungsweise maschinelles Lernen zu einem mächtigen Tool für die Analyse und Vorhersage aus komplexen und großen Datenmengen, sowie als Hilfe für wissenschaftliche Durchbrüche entwickelt.
Oftmals wird maschinelles Lernen als Teilgebiet der künstlichen Intelligenz bezeichnet. Unter maschinellem Lernen versteht man Techniken, mit deren Hilfe ein Verhalten aus Daten gelernt werden kann, anstatt Maschinen statisch zu programmieren. Dabei ist es ein grundlegendes Ziel, neue Beziehungen und Zusammenhänge in den Daten zu erkennen, um anschließend präzise Vorhersagen zu treffen und Wissen in einem verständlichen Weg zu extrahieren [16]. Die verschiedenen Arten des maschinellen Lernens werden in diesem Kapitel erläutert.

3.2 Arten des maschinellen Lernens

Maschinelles Lernen (Abkürzung: ML) inkludiert eine Vielzahl an Algorithmen und Techniken die zur Bildung von datenbasierten Modellen genützt werden können um Probleme zu lösen [31]. Die Grundlage bildet dabei das selbständige Lernen anhand der Basis von Daten, um so die Vorhersage einer Problemstellung zu bestimmen. Tatsächlich geht es bei allen Methoden darum, eine mathematische Funktion $f: X \rightarrow Y$ zu konstruieren, beziehungsweise zu erlernen [37].
In diesem Abschnitt werden die maschinellen Lernmethoden unterteilt. Diese Methoden können in die Kategorien überwachtes Lernen (engl. supervised learning), unüberwachtes Lernen (engl. unsupervised learning) und bestärkendes Lernen (engl. reinforcment learning) eingeteilt werden, wie in Abb. 3: dargestellt.

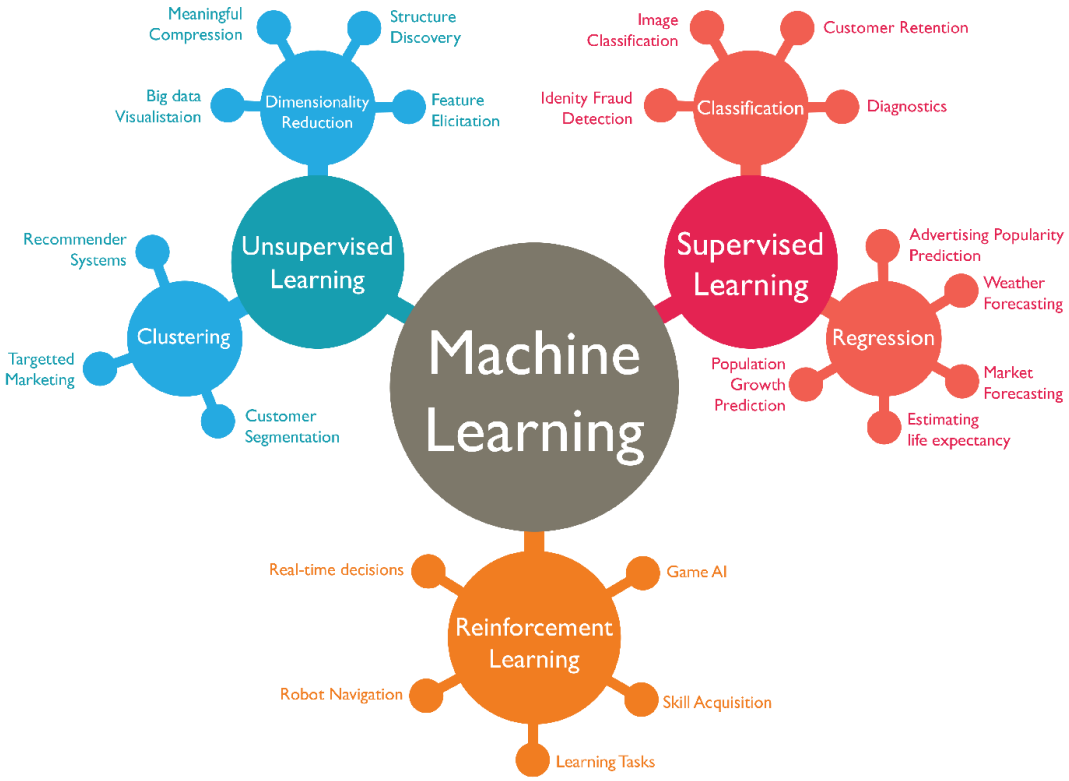

Abb. 3: Machine Learning Übersicht [36]

3.2.1 Supervised Learning

Das Prinzip des Supervised Learning, auch überwachtes Lernen genannt, besteht darin, einen maschinellen Lernalgorithmus mit bekannten Daten, sogenannten Trainingsdaten, dessen gewünschter Output bekannt ist, zu trainieren. Diese Daten müssen genügend viele Werte der Eingabevariablen sowie Ausgangsvariablen beinhalten.
Von diesen Trainingsdaten ist die ‚richtige Antwort' bekannt und in den Daten enthalten. Das Modell, in Abb. 4: in blau dargestellt, versucht anschließend Gesetzmäßigkeiten zu finden, um möglichst zielsichere Voraussagen zu treffen. Dieser Algorithmus wird mit Testdaten überprüft und validiert. Ist die Genauigkeit des Modells zufriedenstellend, können neue, dem Modell noch nicht bekannte, Daten in das Modell geschickt werden und es können Vorhersagen getroffen werden [37].

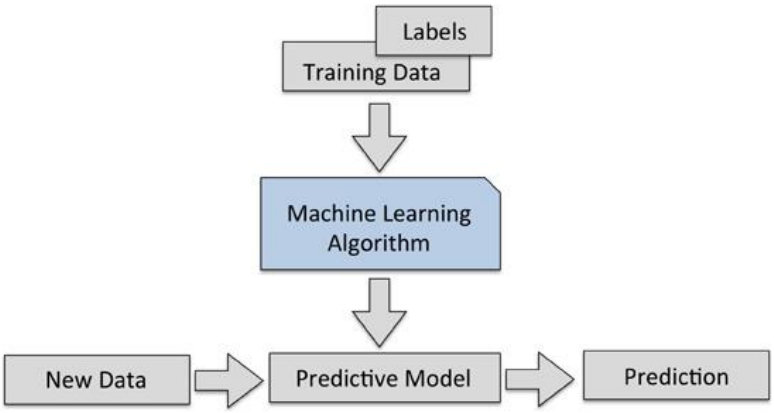

Abb. 4: Supervised Learning Prinzip [11]

Supervised Learning wird in die zwei Kategorien, **Klassifikation** und **Regression** unterteilt. Bei der **Regression** wird eine Funktion ermittelt, die sich möglichst genau an den Datensatz anpasst um den gewünschten Output vorherzusagen. Die Gleichung 2 beschreibt das Prinzip der linearen Regression, wobei $h_\theta(x)$ die Ausgangsvariable darstellt, die vorhergesagt werden soll und $\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$ die einzelnen Signale mit den jeweiligen Gewichtungen θ_0, $\theta_1, \ldots \theta_n$. Die Signale, auch Features genannt, werden als $x_1 \ldots x_n$ bezeichnet [11].

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \quad (2)$$

Zusätzlich zur linearen Regression gibt es noch die nichtlineare Regression, bei der ein Zusammenhang zwischen Eingangs- und Ausgangsgröße anhand einer Kombination von nichtlinearen Funktionen basiert [31].

Das Ziel der **Klassifikation** ist es, ein Modell aufzubauen, das Daten in eine vorgegebene Gruppierung sortieren kann. Die Daten beinhalten unterschiedliche Eigenschaften, die jeweils zu einer Gruppe zugeordnet sind. Das Modell erlernt diese Eigenschaften und kann dann bei neuen Daten die zugehörige Gruppe vorhersagen. Ein Beispiel hierfür wäre die Bilderkennung. Man hat einen Datensatz mit Bildern in denen eine Katze, ein Tisch, ein Glas und ein Hund abgebildet sind. Diese Bilder sind mit dem Namen der abgebildeten Sache markiert und das System erlernt diese 4 Kategorien. Werden dem Modell nun neue Daten gefüttert, kann es diese 4 erlernten Kategorien unterscheiden und richtig zuordnen. Populäre Klassifikationsalgorithmen sind Support-Vektor Maschinen, Neuronale Netze, K-nearest Neighbours und Entscheidungsbäume (engl. Decision Tree) sowie zusammengeschaltete Lernsysteme (engl. ensemble learning) wie zum Beispiel Zufallswald (engl. Random Forest) und Gradient Boosting [31].

3.2.2 Unsupervised Learning

Das Prinzip des Unsupervised Learnings, auch unüberwachtes Lernen genannt, besteht darin, Strukturen in unmakierten Daten zu finden. Im Gegensatz zum Supervised Learning ist hier die ‚richtige Antwort' nicht bekannt. Hier handelt es sich um eine unbekannte

Datenstruktur ohne bekannten Output. Dieser Ansatz wird verwendet um eine **Segmentierung** (engl. Clustering) oder eine **Dimensionsreduktion** (engl. Dimensionality Reduction) anzuwenden [37]. In Abb. 5: ist ein sehr einfaches Beispiel des Clusterings dargestellt. Hierbei versucht das Modell, aus den Daten Muster und Ähnlichkeiten zu erkennen und diese dann zu Gruppieren. Dem Algorithmus wird nicht vorgegeben, ob es sich um einen Apfel, Birne oder um eine Kiwi handelt. Hierbei wird ohne vorheriges Modelltraining oder Datenkenntnis versucht die Daten in Gruppen zu teilen.

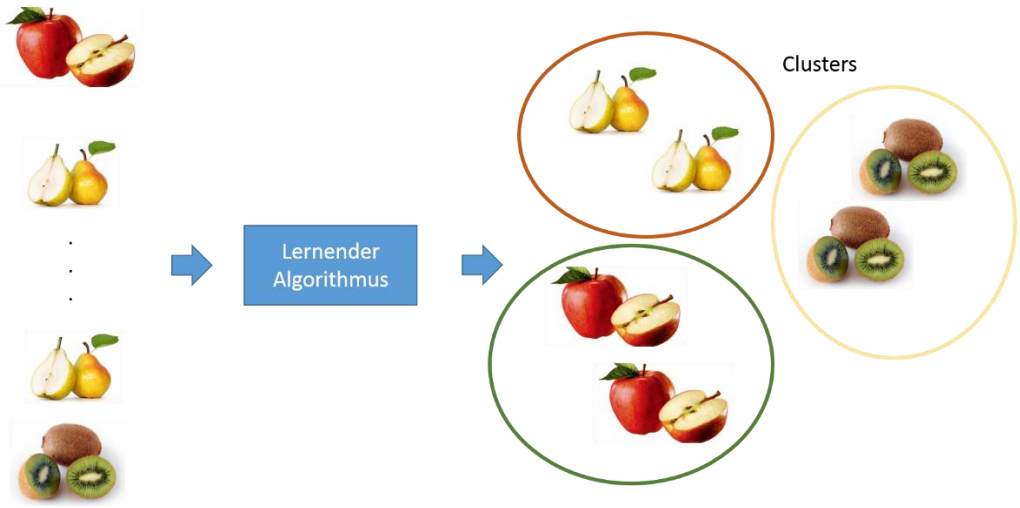

Abb. 5: Clustering Beispiel

Es gibt viele Typen von Clustering-Methoden die laut [37] wie folgt aufgeteilt werden können:

- Hierachische Methoden: agglomerative (group average) und divisive Methoden (Mincut)
- Iterative Methoden: k-means, k-medoid
- Density-based Methoden: DBSCAN
- Stochastische Methoden: gaussian mixture models

Die **Dimensionsreduktion** wird verwendet, wenn eine Unmenge an Features im Datensatz vorhanden ist. Diese Methode reduziert die Anzahl an Features anhand der Auswahl eines repräsentativen Teiles der Features. Einer der berühmtesten Algorithmen dafür ist der Algorithmus „nearest neighbours“ und die Hauptkomponentenanalyse (engl.: „Principal Component Analysis“ - Abkürzung: PCA). Die Varianz der Features wird hier als Kriterium zur Sortierung der Features verwendet [27]. Ziel dabei ist es, die Features zu reduzieren, mit möglichst geringem Informationsverlust. In Abbildung Abb. 6: werden links hierbei dreidimensionale Samples dargestellt, abhängig von den Variablen x_1, x_2 und x_3.

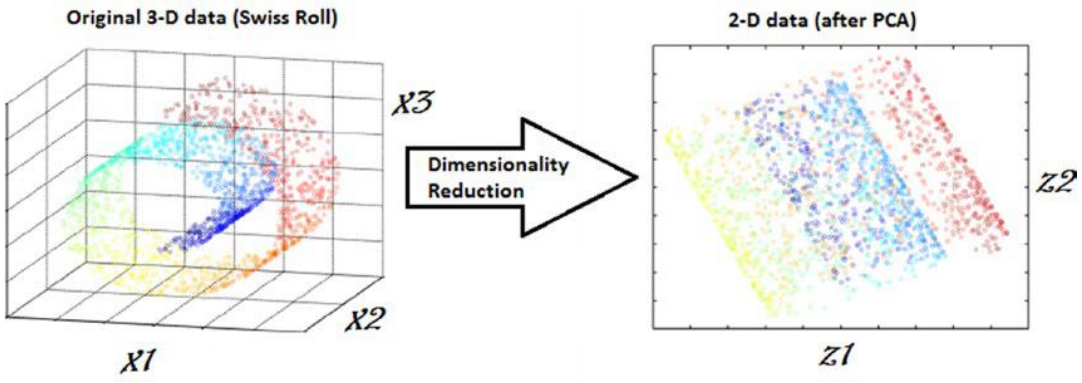

Abb. 6: Dimensionsreduktion aus [30]

Mittels der Hauptkomponentenanalyse wird der Datensatz in eine zwei dimensionale Dimension transformiert. Somit sind die Samples nur mehr von den neu generierten Features z_1 und z_2 abhängig [30]. Dieser Vorgang reduziert die Größe des Datensatzes im Allgemeinen und kann somit als Methode zur Datenkompression gesehen werden. Infolgedessen wird weniger Speicher benötigt und somit die Geschwindigkeit eines maschinellen Lernsystems erhöht [53].

3.2.3 Reinforcement Learning

Dies ist ein Verfahren, auch bestärkendes Lernen genannt, bei dem der Lernalgorithmus, auch Agent genannt, eine Strategie erlernt und somit selbständig durch Versuch und Irrtum zum Ergebnis gelangt. Der Agent wird durch eine Belohnung oder Bestrafung systematisch trainiert. Hierzu wird dem Agent eine positive Belohnung oder eine negative Belohnung, beziehungsweise Bestrafung, nach jeder Aktion vergeben und daraus bildet er eine Entscheidungsstruktur [37]. Wie in Abb. 7: abgebildet, tätigt der Agent eine Aktion, verändert somit die Umgebung und bekommt eine Belohnung zurück. Der Agent muss selbst lernen, was die beste Strategie ist, um auf Dauer die größte Belohnung zu erhalten. Als erstes muss der Agent beobachten und feststellen in welchem Zustand er sich gerade befindet, danach muss er eine Aktion auswählen in dem er die bereits erlernte Strategie anwendet. Anschließend wird je nach neuem Zustand der Umgebung eine Belohnung vergeben und die Strategie bzw. Entscheidungsstruktur aktualisiert.

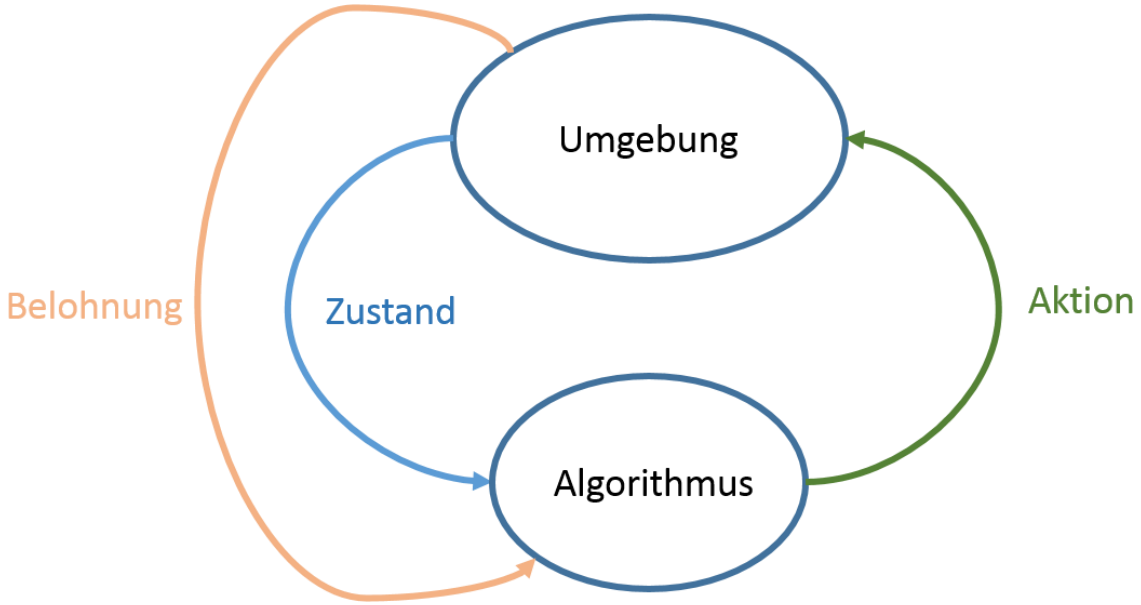

Abb. 7: Reinforcment Learning angelehnt an [18]

Anschließend wird solange iteriert bis eine optimale Strategie gefunden wurde. Ein Beispiel für bestärkendes Lernen ist das Erlernen eines Bewegungsablaufes, wie zum Beispiel das Gehen. Eine der berühmtesten Anwendungen des bestärkenden Lernens, war die übermenschliche Leistung des Computerprogrammes AlphaGo im Spiel Go [17].
Populäre Algorithmen sind der „Markov decision process“, die „Monte-Carlo Methode“, sowie „Temporal Difference Learning“ [18].

3.3 Vorgehensmodell

Um ein ML Projekt erfolgreich abzuschließen sind verschiedene Schritte nötig. Dies startet mit der Datenbeschaffung wie in Abb. 8: dargestellt. Anschließend muss ein Grundverständnis der Daten gebildet und eine Analyse der Daten durchgeführt werden.
Die Aufzeichnung der Daten erfolgt anhand vieler Gründe, maschinelles Lernen spielt dabei nicht unbedingt eine übergeordnete Rolle. Jedes lernende System hat spezifische Anforderungen, welches Datenformat verwendet werden muss, daher müssen die Daten zuerst aufbereitet werden, um diesen Anforderungen zu entsprechen. Weiters kann die Selektion und Aufbereitung der Daten eine große Auswirkung auf die anschließende Modellbildung haben. Aus diesen Gründen ist die Datenaufbereitung einer der wichtigsten und kritischsten Schritte in jedem maschinellen Lernprojekt [53].
Je nach Problemstellung, wird ein Algorithmus zur Modellbildung ausgewählt. Dieses Modell wird anschließend anhand der aufbereiteten Daten trainiert und evaluiert.

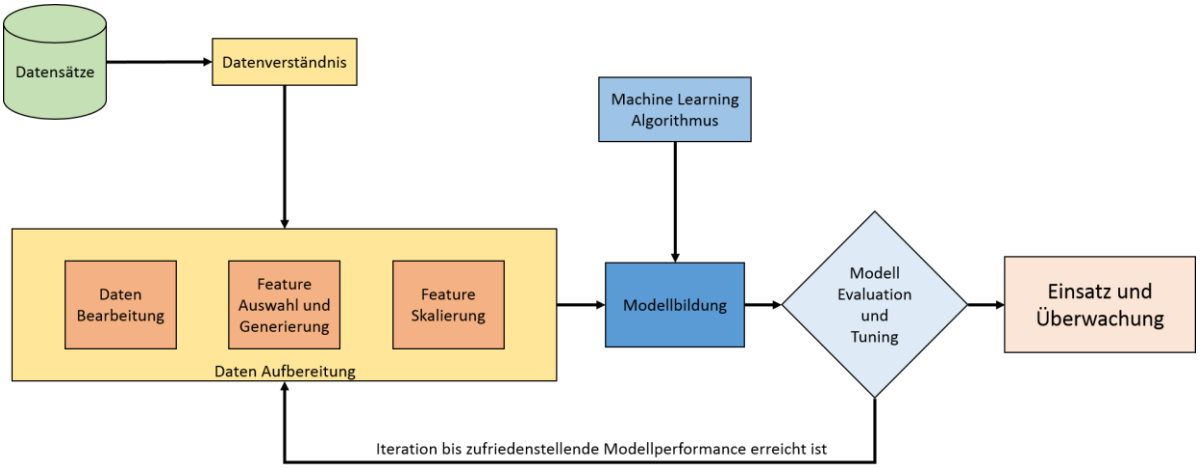

Abb. 8: Übersicht maschinelles Lernen, basierend auf [31]

Dieser Prozess der Datenaufbereitung, Modellbildung und Evaluation wird solange wiederholt bis die Modellperformance zufriedenstellend ist. Die Datenqualität und die Menge an enthaltener, nützlicher Information sind die Grundbausteine für den Erfolg eines datenbasierten ML Modelles [11].

3.3.1 Datenverständnis

Um ein Datenverständnis aufzubauen, werden die unbearbeiteten Rohdaten betrachtet. Zur schrittweisen Erlangung eines Datenverständnisses bildet das CRISP Modell eine gute Grundlage [20]. Dieses Modell basiert auf 4 grundlegenden Schritten, um das Datenverständnis aufzubauen. Der erste Schritt ist die **Datenerfassung**. Hierbei sollen aus allen verfügbaren Datenquellen die Daten gesammelt werden.
Im zweiten Schritt soll die **Datenstruktur beschrieben** werden. Hierbei sollen die groben, oberflächlichen Eigenschaften der Daten erkundet werden. Die Qualität und Quantität der Daten soll analysiert werden, sowie das Datenformat und die Dimension der Datensätze. Es soll überprüft werden, ob die vorhandenen Daten die relevanten Anforderungen des maschinellen Lernproblems erfüllen.
Im dritten Schritt sollen die **Daten erforscht und tiefgehend** untersucht werden. Hierbei kann mittels Statistik ein Überblick der jeweiligen Attribute bzw. Features geschaffen werden, indem man beispielsweise das Minimum, Maximum, den Durchschnitt und die Standardabweichung betrachtet [19]. Hierbei wird ersichtlich, welche Wertebereiche abgedeckt werden und folglich kann dies auf Plausibilität geprüft werden. Durch eine Korrelationsanalyse soll der Zusammenhang zwischen den Features betrachtet werden. Die meist verbreitete Korrelation ist die Pearson Korrelation [21]. Hierbei wird die lineare Beziehung zwischen zwei Variablen gemessen. Der Korrelationskoeffizient kann Werte zwischen -1 und +1 annehmen, wobei -1 eine negative Korrelation, 0 keine Korrelation und +1 eine positive Korrelation darstellt [40]. Weiters sollen mögliche Probleme erörtert werden, die hinsichtlich der Daten entstehen könnten.
Im letzten Schritt soll die **Datenqualität** verifiziert werden. Die Daten sollen auf Vollständigkeit geprüft werden. Fehlende Werte, leere Features, sowie die Plausibilität der Features soll

überprüft werden. Es soll der Datentyp für jedes Attribut bestimmt werden. Hierbei kann es sich um numerische oder kategorische Daten handeln. Die Daten können in verschiedensten Formaten vorliegen. Bei Echtzeit Datensätzen handelt es ich oftmals um Zeitreihen. Diese bringen bestimme Eigenschaften mit, die im Folgenden erklärt werden.

Zeitreihen

Eine Zeitreihe ist eine Sequenz *S* von historischen Messungen y_t von einer Variablen *y* in bestimmten Zeitabständen. Diese Zeitabstände können je nach Messung variieren und hängen von der Abtastrate ab.
Zeitreihen haben bestimmte Eigenschaften und sind im Unterschied zu einer normalen Tabelle sehr verschieden. Sie sind abhängig von der Zeit, die Reihenfolge der Signale spielt eine große Rolle und bei mehreren Versuchen kann die Länge der Zeitreihe unterschiedlich sein [49].

3.3.2 Datenaufbereitung

Die Datenaufbereitung gliedert sich in drei Schritte, der erste Schritt ist die **Datenbearbeitung**. Anschließend werden die **Features ausgewählt** und neue Features generiert. Der nächste Schritt ist die **Skalierung** der Features. Abschließend werden diese noch in **Trainings- und Testdaten aufgeteilt**.
Das benötigte Endformat um einen maschinellen Lernalgorithmus anzuwenden, besteht aus Eingangsvariablen $x_1 \dots x_m$ und einer Ausgangsvariable y wie in Abb. 9: abgebildet. Jedes Beispiel an dem der Algorithmus gelernt oder getestet werden kann, wird als Sample bezeichnet. Somit gibt es eine Anzahl an n Samples und m Features bzw. Eingangsgrößen.

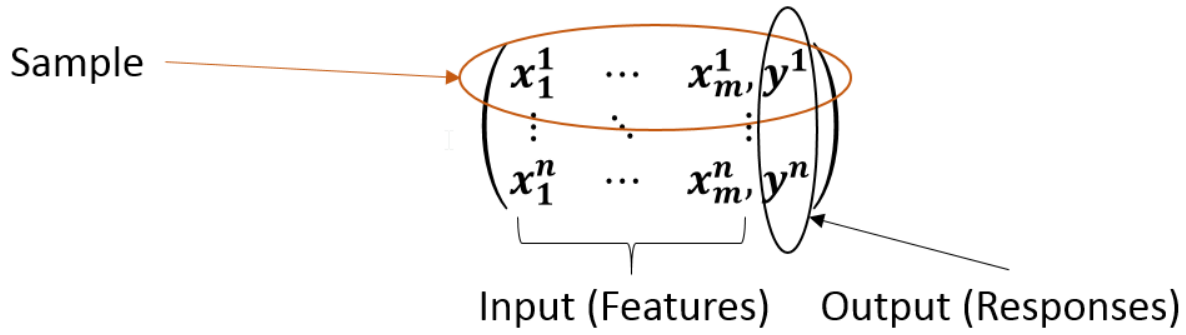

Abb. 9: Datenformat maschinelles Lernen [22]

3.3.2.1 Datenbearbeitung

Die Datenbearbeitung soll die Rohdaten in ein Datenformat transformieren, um maschinelle Lernmethoden anwenden zu können. In der Datenbearbeitung soll zuerst eine Selektion der Daten vorgenommen werden. Es soll entschieden werden welche Daten als Basis für das maschinelle Lernen eingesetzt werden. Diesen Vorgang nennt man **Diskretisierung.** Es wird dabei versucht, die quantitative Anzahl an Daten zu reduzieren, um somit einen qualitativen

Datensatz herzustellen [53]. Hierbei soll die Relevanz der Daten, die Datenqualität, das Datenformat, sowie die Quantität als Entscheidungsgrundlage dienen [20].
Manche maschinelle Lernsysteme benötigen numerische Daten. Daher sollen kategorische Daten in numerische Daten konvertiert werden [53].
Der nächste Schritt ist die Datenbereinigung. Fehlende Daten können zu Problemen während der Analyse führen. Es können **fehlende Werte** in den Daten auftreten, sowie sichtlich falsche Daten, sogenannte **Ausreißer** (engl. Outliers). Die erste Möglichkeit die Daten zu bereinigen ist, die Zeilen in denen Werte fehlen, zu löschen. Dies macht Sinn, wenn in einem insignifikanten Anteil an Zeilen (z.B.: kleiner 5%) Werte fehlen [35]. Die zweite Möglichkeit ist, die fehlenden Werte zu ersetzen. Hierzu gibt es mehrere Varianten. Es können fehlende Werte mit dem Mittelwert, Minimum oder Maximum ersetzt werden. Weiters können diese fehlenden Werte durch lineare oder nichtlineare Interpolation ersetzt werden.
Diese Ausreißer können aufgrund von Messfehlern, menschlichen Eingabefehlern, Datenverarbeitungsfehlern oder Instrumentenfehlern entstehen.
Um Ausreißer zu detektieren können statistische Tests, dichtebasierte Algorithmen (z.B.: LOF – engl.: local outlier factor) oder deviationsbasierte Ansätze verwendet werden [23] und [24].
Das Datenformat soll in ein Format umgewandelt werden, das mittels der Machine Learning Umgebung einlesbar und bearbeitbar ist. Wenn nötig, soll eine Dimensionsreduktion angewendet werden, wie in Kapitel 3.2.2 bereits erläutert.

3.3.2.2 Feature Auswahl und Generierung

Die Vorhersagegüte eines maschinellen Lernmodelles wird durch die Qualität des Inputs, also der Datenqualität, bestimmt. Einer der wichtigsten Aspekte im ML ist die Feature Auswahl und die Feature Generierung. Je mehr Features zur Modellbildung verwendet werden, desto länger dauert die Modellbildung. Werden weniger Features verwendet, desto einfacher ist es mögliche Kausalitäten aus dem Modell abzuleiten [26].
Diese **Feature Auswahl** bzw. Reduktion kann aufgrund statistischer Verfahren oder, je nach Problemstellung, von Expertenteams bestimmt werden. Zur statistischen Bestimmung stehen mehrere Verfahren zur Verfügung. Hierbei kann eine Hauptkomponentenanalyse (engl. Principal Component Analysis – short: PCA) verwendet werden, wie in Kapitel 3.2.2 erläutert. Weiters gibt es noch Filter Methoden, Wrapper Methoden und Embedded Methoden wie in [31] erläutert. Zur Feature Auswahl gibt es noch weitere Methoden, wie heuristische Methoden oder stochastische Methoden, die in [28] ausführlich erklärt sind.
Unter **Feature Engineering** versteht man das Generieren neuer Features anhand der bereits vorhandenen Daten. Dies kann eine Summe oder durch eine Kombination mehrerer Features gebildet werden. Feature Generierung ist ein zeitaufwändiges, schwieriges und langsames Vorgehen das meist Expertenwissen voraussetzt um informationsreiche Features zu erzeugen [29]. Hierbei werden keine neuen Daten hinzugefügt, sondern die Datenstruktur und der Inhalt werden anhand der neu generierten Features besser und umfangreicher beschrieben.
Der grundlegende Unterschied zwischen der Feature Auswahl und der Feature Generierung ist das Endresultat. Angenommen wir haben Features F_1, F_2, F_3, F_4 und in beiden Ansätzen werden im Endresultat 2 Features verwendet. Bei der Feature Auswahl wird eine Teilmenge der Features verwendet, zum Beispiel F_1 und F_3. Bei der Feature Generierung werden diese

2 Features aus einer Kombination der 4 original Features gebildet. Dies kann zum Beispiel anhand von $F_1' = \sum a_i * F_i$ und $F_2' = \sum b_i * F_i$, wobei a_i und b_i Konstanten sind, gebildet werden [53].
Ein neuartiger Ansatz ist das **Feature Learning**, in dem ein Algorithmus automatisch Features generiert. Hierbei werden Beziehungen in den Daten erörtert und mathematische Funktionen angewendet um neue Features zu erstellen [30].

3.3.2.3 Feature Skalierung

Die meisten Machine Learning Algorithmen profitieren von einer Skalierung der Features. Weit verbreitet sind zwei verschiedene Ansätze: „Normalisierung“, „Standardisierung“. Meistens wird unter der Normalisierung eine Skalierung der Features in einem Bereich zwischen $[0,1]$ verstanden und diese wird in Gleichung 3 beschrieben. Hierbei ist x_j ein Wert in der Feature Spalte, x_{min} der minimale Wert und x_{max} der größte Wert dieses Features. Der Ansatz „Standardisierung“ wird in Gleichung 4 beschrieben. Durch diese Standardisierung wird eine Standard Normalverteilung erreicht, indem der Mittelwert um den Wert 0 zentriert wird und die Standardabweichung bei 1 liegt [11].

$$x'_{jnorm} = \frac{x_j - x_{min}}{x_{max} - x_{min}} \quad (3)$$

$$x'_j = \frac{x_j - \mu_j}{\sigma_j} \quad (4)$$

Um zum Beispiel das j –te Feature zu skalieren, wird der Mittelwert μ_j abgezogen und durch die Standardabweichung σ_j dividiert [33].
Weitere ähnliche Methoden sind die Min-Max Skalierung oder die Robuste Skalierung, wie in [31] ausführlich anhand von Gleichungen und Code Beispielen erklärt.

3.3.2.4 Aufteilung in Trainings- und Testdaten

Der letzte Schritt der Datenaufbereitung ist die Aufteilung des Datensatzes in Trainings-, Validierungs- und Testdaten, wie in Abb. 10: gezeigt. Ein Validierungsdatensatz soll vor allem dann verwendet werden, wenn die Datenanzahl sehr hoch ist [34]. Der Trainingsdatensatz wird zur Bildung des Modelles verwendet, um zum Beispiel in einem linearen Modell die Gewichtungen zu bestimmen. Hierbei erlernt das Modell die grundlegende Struktur der Daten um die Ausgangsvariable zu bestimmen [37].
Der Validierungsdatensatz kann hierbei mehrere Funktionen einnehmen. Diese Validierungsdaten können dazu verwendet werden, um die Parameter eines Modelles zu optimieren [38]. Laut [11] kann dieser Validierungsdatensatz auch zur Modell Selektion verwendet werden. Hierbei werden mehrere Modelle anhand der Trainingsdaten trainiert und anschließend aufgrund der Prognosegüte der Validierungsdaten ausgewählt. Ein Nachteil davon ist, dass die Prognosegüte sehr abhängig davon sein kann, welcher Teil der Daten in den Teildatensätzen landet. Daher wird später in Abschnitt 3.3.4.3 auf eine robustere Methode, die Kreuzvalidierung, eingegangen.

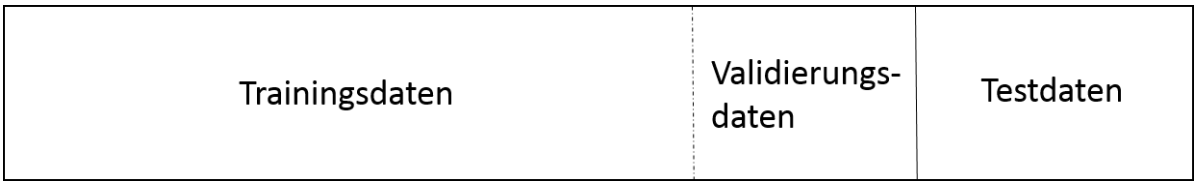

Abb. 10: Aufteilung der Daten in Trainings-, Validierungs- und Testdaten, angelehnt an [37]

Der Testdatensatz ist unabhängig von den Trainingsdaten. Anhand des Testdatensatzes wird schlussendlich die Prognosegüte bewertet. Dies sind Daten die das Modell vorher noch nicht gesehen hat [37].
Laut [34] ist es schwierig, eine allgemeine Regel für die Aufteilung in diese drei Teile aufzustellen. Eine mögliche Aufteilung könnte 50% Trainings-, 25% Validierungs- und 25% Testdaten sein. Wird kein Validierungsdatensatz verwendet, liegt die typische Aufteilung bei ca. 70-85% Trainingsdaten und 15-30% an Testdaten [37].

3.3.3 Modellbildung

Ein Modell repräsentiert in einfachen Worten den Zusammenhang zwischen Eingangs- und Ausgangsvariablen bzw. Features. Hierbei wird ein Algorithmus ausgewählt und anhand von Trainingsdaten ein Modell gebildet. Die Auswahl des Lernalgorithmus hängt vom Datensatz und vom zu lösenden Problem, wie in Kapitel 3.2 erklärt, ab. In diesem Abschnitt wird näher auf die Bildung von Regressions Modellen eingegangen, wie in Kapitel 3.2.1 bereits erläutert. Am besten ist es, mit dem einfachsten Modell zu beginnen. Unter Anwendung eines einfachen Algorithmus, ist das gebildete Modell einfacher verständlich, die Trainingszeit kürzer, sowie die Fähigkeit der Generalisierung des Modelles besser [19].
Regressions Modelle können in drei Hauptkategorien eingeteilt werden [31]:

a) Einfache lineare Regression: Dies ist die einfachste Form eines Regressionsmodelles. Hierbei kann nur eine lineare Beziehung zwischen einer Eingangs- und einer Ausgangsvariablen abgebildet werden.
b) Mehrfache lineare Regression: Dies ist eine Erweiterung von a), da mehrere Eingangsvariablen inkludiert werden können
c) Nichtlineare Regression: Hierbei können auch nichtlineare Beziehungen zwischen den Eingangs- und Ausgangsvariablen abgebildet werden.

Die mathematische und statistische Vorgangsweise der Modellbildung wird in [32] ausführlich erläutert.
Um ein Regressionsproblem zu lösen, stehen mehrere Algorithmen zur Verfügung. Die weit verbreitetsten sind lineare Modelle, Entscheidungsbäume, Support Vektor Maschinen und Neuronale Netze. Im folgenden Abschnitt wird zuerst ein lineares Modell, die Lineare Lasso Regression erklärt. Anschließend wird auf ein nichtlineares Modell, den Zufallswald, eingegangen.

3.3.3.1 Lineares Modell

Ein einfaches lineares Modell wurde mit Gleichung 2 im Kapitel 3.2.1 bereits beschrieben. Das Ziel eines linearen Modelles ist es, die Eingangsvariablen X so zu gewichten, dass die Ausgangsvariable Y möglichst genau vorhergesagt werden kann. Ein Spezialfall der linearen Regression ist die **lineare Lasso Regression**. Lasso bedeutet ausgeschrieben „least absolute shrinkage and selection operator“ [12]. Hierbei wird eine Selektion der Variablen, sowie eine Regulierung der Gewichtungen angewandt. Wichtig hierbei ist der Regulierungswert α. Dieser Wert bestimmt die Regulation, der in Gleichung 2 erwähnten Gewichtungen $\theta_0,\ \theta_1, \dots\ \theta_n$. Je größer α, desto weniger Features werden zur Modellbildung herangezogen. Dies bewirkt, dass mehr Gewichtungen der Features $\theta_1 x_1 + \cdots +\ \theta_n x_n$ zu Null werden. Somit werden nur die wichtigsten Features zur Modellbildung berücksichtigt. Die Vorhersagegenauigkeit kann durch schrumpfende oder null werdende Gewichtungen verbessert werden. Durch diese wird die Verzerrung (engl. bias) vergrößert, jedoch die Varianz (engl. variance) der Vorhersage verringert und dadurch die Gesamtvorhersage Genauigkeit erhöht [13]. Die Verzerrung gibt die Anpassung des Modelles an den Datensatz an, wobei der optimale Fall eine möglichst geringe Verzerrung ist. Die Varianz ist gering, wenn das Modell auf verschiedene Datensätze eine konstante Vorhersagegenauigkeit aufweist [37]. Der genaue Algorithmus für das Finden der Lösung durch das lineare Lasso Modell wird in [13] in Kapitel 6 ausführlich erläutert.

3.3.3.2 Zufallswald (engl. Random Forest)

Um einen Zufallswald (engl. Random Forest [41]) erklären zu können, muss vorerst auf die einfachere Variante des Entscheidungsbaumes eingegangen werden.
Diese erlernen eine hierarchische Folge von Ja/Nein Fragen, die schlussendlich zu einer Entscheidung führen [43]. Die Auswertung eines Entscheidungsbaumes beginnt an der Wurzel (engl. root), wie in Abb. 11: exemplarisch dargestellt. Danach führt dies zu einem Knoten (engl. node) an dem eine Entscheidung getroffen wird [39]. Am Ende steht ein Blatt (engl. leaf), an dem keine Entscheidung mehr fällt, sondern ein Regressions- oder Klassifikationswert ausgegeben wird [37].

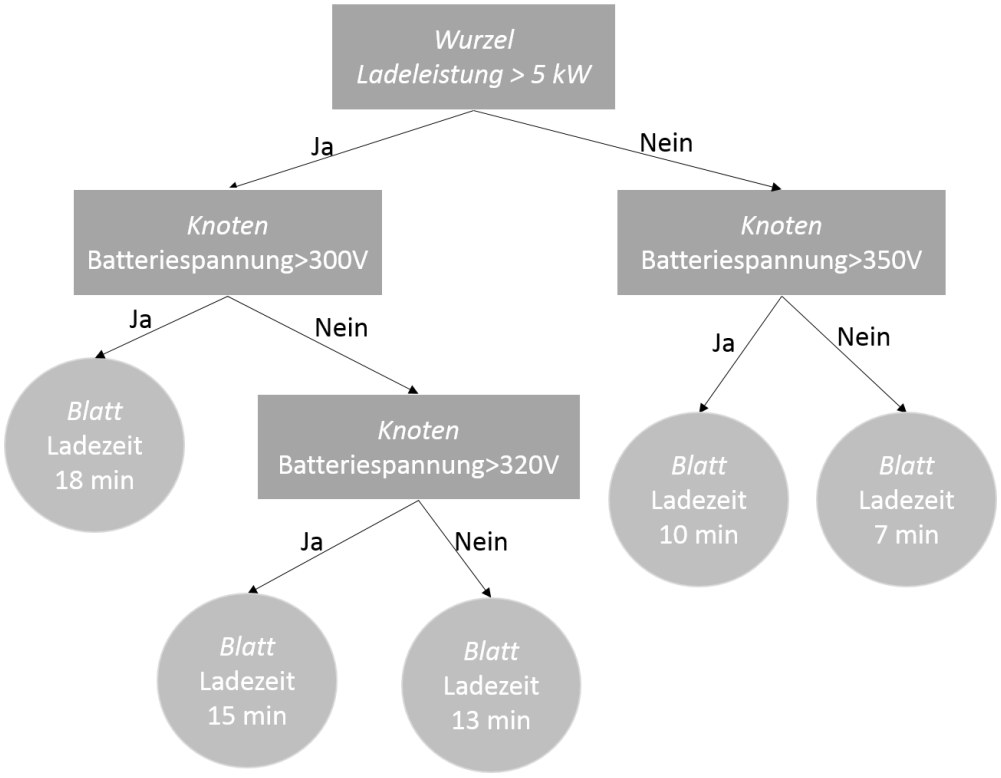

Abb. 11: Entscheidungsbaum

In diesem Beispiel ist ein sehr einfacher Entscheidungsbaum aufgezeigt. Hierbei werden nur zwei Features, Ladeleistung und Batteriespannung, beispielhaft verwendet um das Prinzip eines Entscheidungsbaumes zu erklären. Das Endresultat hängt von der Höhe der Batteriespannung ab und ob mit einer Leistung unter oder über 5 kW geladen wurde. Das Ergebnis ist jeweils die Ladezeit in Minuten.

Der maschinelle Lernalgorithmus bildet diesen Entscheidungsablauf dabei selbst, aufgrund der zugeführten Daten. Dazu gibt es mehrere Arten von Algorithmen. Die drei bekanntesten sind ID3, CART und C4.5. Um einen genauen Einblick in diese Algorithmen zu bekommen, siehe [35] und [43].

Wenn mehrere verschiedene Lernmodelle zusammengeschaltet werden, wird dies **Ensemble Learning** genannt [37].

Ein **Zufallswald (engl. Random Forest)** besteht aus mehreren Entscheidungsbäumen, die zufällig generiert werden. Hierbei werden mehrere Entscheidungsbäume bzw. Lernmodelle zusammengeschaltet und daher zählt der Zufallswald zu den Ensemble Lernern. Bei der Bildung eines Zufallswaldes kommen zwei Methoden zur Anwendung: **Bagging** und **Random Subspace** [53].

Im Folgenden wird die Bagging Methode beschrieben. Es wird für jeden Baum zufällig eine Stichprobe aus den Daten gezogen. Diese Stichprobe beinhaltet eine festgelegte Anzahl an Variablen die auch zufällig ausgewählt werden. Diese Stichprobe dient zur Bildung eines Entscheidungsbaumes und wird anschließend wieder zu den ursprünglichen Daten hinzugefügt (zurücklegen). Zur Bildung des nächsten Entscheidungsbaumes wird wieder eine Stichprobe, wie oben beschrieben, aus den Daten gezogen und ein Baum gebildet [16]. Die Anzahl der Bäume wird vorher definiert und jeder Baum liefert eine Vorhersage der

Ausgangsgröße. Diese Vorhersage der verschiedenen Bäume wird gemittelt und somit ergibt sich die Gesamtvorhersage des Zufallswaldes. Diese Methode, mehrere Vorhersagemodelle $M_1, M_2, \ldots M_n$ anhand von zufällig gezogenen Stichproben des Originaldatensatzes zu trainieren und anschließend den Mittelwert der Vorhersage aller n Modelle zu bilden, um eine genauere Vorhersage zu bekommen, nennt man Bagging. In Abb. 12: ist das Prinzip des Bagging grafisch dargestellt. Es werden aus den Daten zufällig n Stichproben (engl. bootstrap samples) gezogen und anhand dieser Stichprobe ein Modell gebildet. Jedes Modell liefert eine Vorhersage und durch das Zusammenführen (engl. bootstrap aggregation) wird eine Gesamtvorhersage gebildet. Dies kann mittels einer Mehrheitsabstimmung (engl. majority vote) oder der Durchschnittsbildung erfolgen.

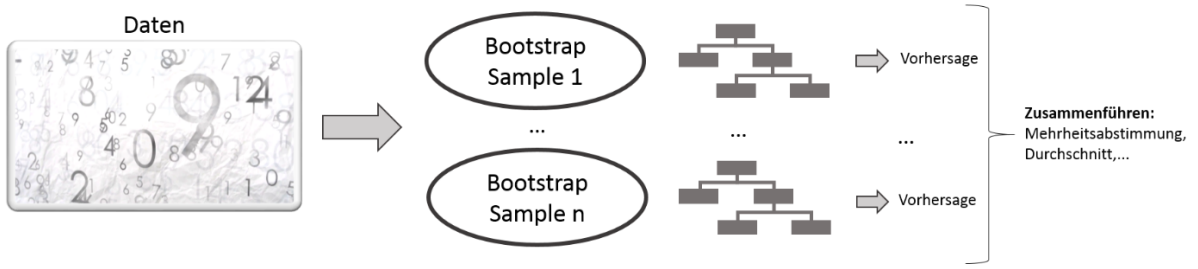

Abb. 12: Bagging angelehnt an [14]

Dabei wird die Vorhersagegenauigkeit erhöht, die einfache und leicht interpretierbare Struktur geht jedoch verloren [14]. Diese Methode wird bei der Bildung von mehreren Bäumen in einem Zufallswald verwendet. Bagging kann dabei für jegliche Kombination von Modellen verwendet werden, nicht nur bei Entscheidungsbäumen.

Die Random Subspace Methode (Abkürzung: RSM) soll dazu dienen, die Diversität zwischen den Teilmengen zu erhöhen und die Korrelation zwischen den Schätzern zu erniedrigen. Hierbei wird eine zufällig ausgewählte Teilmenge an Features zur Bildung der Modelle verwendet [15]. Diese Teilmenge n wird zufällig aus der Gesamtmenge N an Features gezogen, wobei $n < N$ ist. Aus empirischen Studien wird als Daumenregel $n = \frac{N}{2}$ empfohlen [52].

Zur Bildung eines Entscheidungsbaumes in einem Zufallswald werden beide Methoden, RSM und Bagging, verwendet. Jeder Entscheidungsbaum wird von einem bootstrap sample des originalen Datensatzes gebildet. An jedem Zweig, wird eine zufällige Teilmenge n der Gesamtmenge an N Features des bootstrap Samples zur Aufspaltung in einen Knoten herangezogen. Hierbei wird $n = \log_2(N + 1)$ empfohlen [52]. In einem normalen Entscheidungsbaum wird die Aufteilung in einen Knoten anhand der besten Aufteilung aller Variablen gebildet. Bei einem Zufallswald wird die beste Teilmenge der Variablen, die zufällig ausgewählt wurde, zur Aufspaltung verwendet [45].

Da jeder Baum einen Beitrag zur Vorhersage des Zufallswaldes liefert, ist es bei vielen Bäumen relativ unübersichtlich zu erkennen, wie eine Vorhersage gebildet wurde. Zum einfacheren Verständnis, welche Features zur Bildung des Ergebnisses wichtig waren, kann die Wichtigkeit der Variablen ausgegeben werden. Die Bildung dieser Wichtigkeit ist in [44] sehr deutlich und anschaulich erklärt.

3.3.4 Modellevaluation

Die Modellevaluation von Regressionsproblemen kann mittels verschiedenster Metriken bewertet werden. Unter anderem mittels dem mittleren absoluten Fehler (engl.: mean absoulte error), der mittleren quadratischen Abweichung (engl.: mean squared error) oder auch dem mittleren Abweichungsquadrat (engl.: root mean square error) .
Zur Evaluierung eines Klassifikationsproblems kann eine Wahrheitsmatrix oder die Fläche unter der Kurve (engl. area under curve – AUC) berechnet werden [52].

3.3.4.1 Modellevaluation Regression

Bestimmtheitsmaß

Eine sehr weit verbreitete Methode, um die Modelle untereinander zu vergleichen, ist das Bestimmtheitsmaß R^2. Die Berechnung wird wie in Gleichung 5 erläutert.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = \frac{erklärte\ Streuung}{gesamte\ Streuung} \qquad (5)$$

Der R^2 Wert bewegt sich im Bereich von $0 \leq R^2 \leq 1$. Je größer der Wert von R^2 desto näher passt sich die Vorhersage an die echten Werte an [50].

Root Mean Square Error

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Vorhersage_i - Soll\ Wert_i)^2}{N}} = \sqrt{MAE^2} \qquad (6)$$

Die Wurzel aus der mittleren quadratische Abweichung (engl. Root Mean Square Error, Abkürzung: RMSE) ist die Wurzel aus dem Quadrat des mittleren absoluten Fehlers (engl. mean absolute error), wie in Gleichung 6 erläutert. Der RMSE liefert eine Aussage über die Prognosegüte eines Modelles. Der Bereich kann von 0 bis ∞ variieren und ist unabhängig von der Richtung des Fehlers. Je kleiner der RMSE Wert, desto besser nähert sich das Modell den Daten an [54]. Ein wichtiger Unterschied zwischen RMSE und MAE ist, dass durch das Quadrieren eine hohe Gewichtung auf große Abweichungen gegeben wird [51].

3.3.4.2 Modellevaluation Klassifikation

Zur Modellevaluation der Klassifkation, kann eine Wahrheitsmatrix herangezogen werden. Dazu werden alle möglichen Fälle in eine wahre und falsche Klasse unterteilt, wie in Abb. 13: ersichtlich. Gibt es zwei mögliche Klassen, positiv und negativ, gibt es hierzu 4 verschiedene Kombinationen. Wird vom Modell Positiv vorhergesagt und der echte Wert ist auch positiv, ist dies **richtig positiv** (engl. true positive). Sagt das Modell Positiv voraus, der echte Wert ist aber negativ, somit ist dies **falsch positiv** (false positive). Wird vom Modell Negativ vorhergesagt, und der echte Wert ist positiv, ist dies **falsch negativ** (engl. false negative).

Wird vom Modell Negativ vorhergesagt und der echte Wert ist auch negativ, somit ist dies **richtig negativ** (engl. true negative). Gewünscht werden richtig vorhergesagte Klassen, also richtig positive und richtig negative Vorhersagen [52].

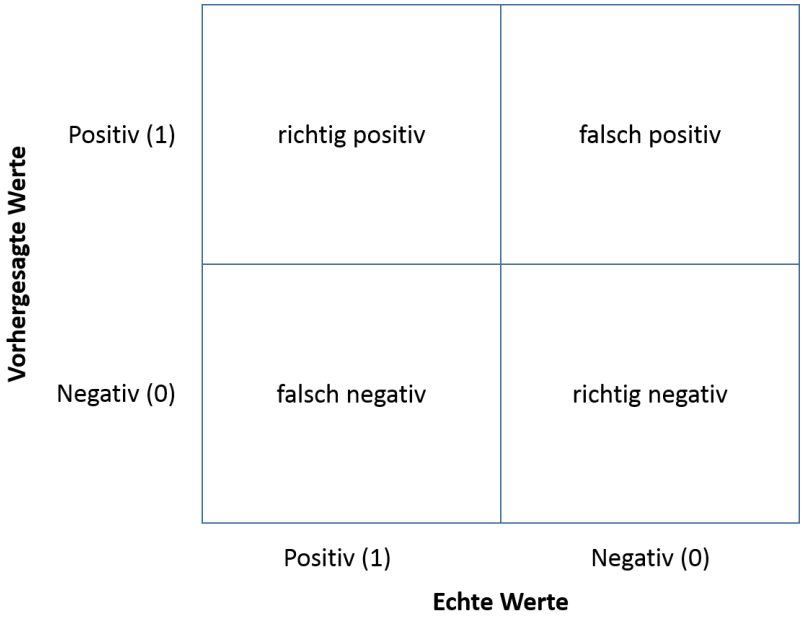

Abb. 13: Wahrheitsmatrix angelehnt an [52]

Die Genauigkeit für die jeweilige vorhergesagte Klassen wird wie folgt in Gleichung 7,8,9 und 10 beschrieben, berechnet [53].

$$Positiver\ Vorhersagewert = \ Genauigkeit = \frac{richtig\ positiv}{richtig\ positiv + falsch\ positiv} \tag{7}$$

$$Rate\ richtig\ positiv = \frac{richtig\ positiv}{richtig\ positiv + falsch\ negativ} \tag{8}$$

$$Rate\ richtig\ negativ = \frac{richtig\ negativ}{richtig\ negativ + falsch\ positiv} \tag{9}$$

$$Negativer\ Vorhersagewert = \frac{richtig\ negativ}{richtig\ negativ + falsch\ negativ} \tag{10}$$

3.3.4.3 Kreuzvalidierung

Kreuzvalidierung ist eine Methode um den kompletten Datensatz abzugrenzen und gegeneinander abzutesten. Jeder Datenpunkt muss hierbei einmal als Testdatenpunkt verwendet werden. Die klassische Ausführung der Kreuzvaliderung ist die k Fold Kreuzvalidierung. Zuerst wird der Datensatz in k gleiche bzw. annähernd gleiche Datensätze aufgeteilt. Eine übliche Aufteilung der Trainings- zu Testdaten ist 80% zu 20%, was einem k

von 5 entspricht. In Runde 1 werden die ersten 20% als Testdatensatz gewählt und mit den restlichen Daten wird das Modell trainiert, wie in Abb. 14: ersichtlich.

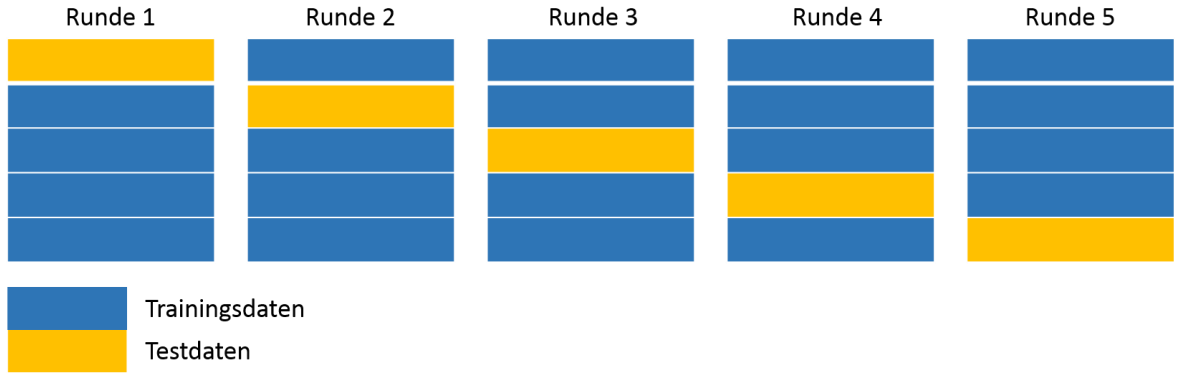

Abb. 14: Kreuzvalidierung, angelehnt an [52]

Die blau markierten Blöcke bilden den Trainingsdatensatz mit dem das Modell trainiert wird und der gelb markierte Block bildet den Testdatensatz um die Prognosegüte des Modelles zu validieren. Anschließend werden in den folgenden Runden nacheinander der zweite, dritte, vierte und fünfte Teil der Daten als Testdatensatz verwendet. Der Rest wird jeweils als Trainingsdatensatz des Modelles herangezogen. Anschließend wird jedes Modell aufgrund der Prognosegüte bewertet und das Modell mit dem geringsten Prognosefehler wird als endgültiges Modell verwendet [51].

3.3.4.4 Bias und Varianz

Ein grundlegendes Problem bei überwachtem Lernen, ist der Kompromiss zwischen Bias und Varianz. Im optimalen Fall sollte das Modell zwei Eigenschaften erfüllen [25]:

- Es soll sensibel sein, um die Muster des Trainingsdatensatzes genau zu erfassen
- Es soll gute Generalisierungseigenschaften haben, um auch auf noch nicht gesehen Daten eine gute Prognosegüte zu liefern

Im Allgemeinen findet immer ein Kompromiss zwischen diesen beiden Eigenschaften statt. Dies kann als Kompromiss zwischen Bias und Varianz gesehen werden [53], wie in Abb. 15: gezeigt.
Unter Bias, auch Verzerrung genannt, versteht man einen systematischen Fehler [37]. Von hohem Bias spricht man, wenn die Prognosegüte anhand der Trainings- und Testdaten gering ist. Das heißt, dass sich das Modell nicht gut an die Trainingsdaten anpasst, sowie auch die Prognosegüte anhand der Testdaten gering ist [25]. Dies nennt man Under-fitting.

Unter hoher Varianz versteht man eine hohe Prognosegüte des Trainingsdatensatzes, jedoch eine sehr geringe Güte anhand des Testdatensatzes [25]. Das Modell ist sozusagen überangepasst an die bereits gesehenen Daten, generalisiert jedoch schlecht auf die ungesehenen Daten [26]. Dies nennt man Over-fitting.

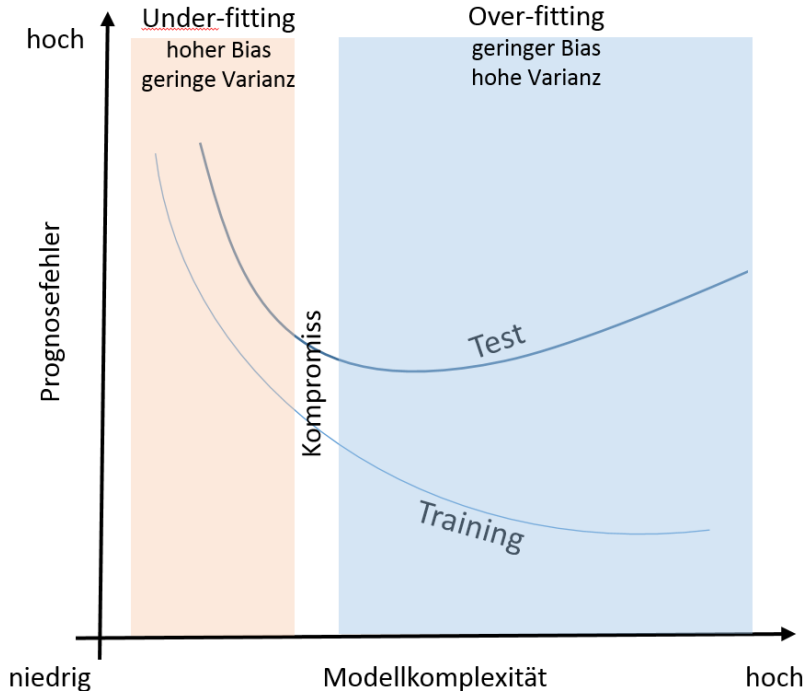

Abb. 15: Under-fitting und Over-fitting, angelehnt an [25]

Das Ziel ist es, eine Balance zwischen Under-fitting und Over-fitting zu finden um auf beide Datensätze, Test und Training, einen möglichst geringen Prognosefehler zu erreichen.
Over-fitting kann mittels Reduzierung der Features, Dimensionsreduktion oder durch Vergrößerung des Datensatzes erreicht werden. Under-fitting kann durch repräsentativere Daten und optimalere Modellparameter vermieden werden [25].

3.3.5 Modelltuning

Die bisherig erklärte Modell Bildung erfolgte anhand von Standardparametern der Modelle. Der iterative Prozess, den Datensatz anhand von Feature Generierung und Selektion beziehungsweise die Modellparameter (sogenannte Hyperparameter) zu verändern, bildet den Kern des Modelltunings. Diese Parameter werden vor der Modellbildung gesetzt und anschließend wird das Modell evaluiert [31].
Hierzu gibt es mehrere Strategien [26]:

- Grid Search: Hierbei wird ein Raster an Parametern getestet und evaluiert. Für jede mögliche Kombination der Parameter wird ein Modell trainiert und getestet. Dies ist je nach Modell und Größe der Daten sehr zeitaufwändig und rechenintensiv. Anschließend wird die Prognosegüte jedes Parametersets bewertet und die beste Kombination zurückgegeben.
- Randomized Grid Search: Bei sehr großen Datensätzen und komplexen Modellen ist es sinnvoll, das Modell mittels zufällig ausgewählten Parametersätzen zu trainieren und zu testen. Die Qualität der Ergebnisse ist dabei ähnlich gut wie bei der Methode Grid Search. Wieder wird die beste gefundene Parameterkombination zurückgegeben.

4 Anwendungsfall: Ladezeitprognose mittels maschinellem Lernen

Wie in Kapitel 2 erläutert, ist das Laden, die Reichweite und die Dauer des Ladens der Batterie eine wichtige Kernthematik der Elektromobilität. Diese drei Faktoren werden die Akzeptanz, sowie das Kaufverhalten der Kunden hinsichtlich der Elektrofahrzeuge stark beeinflussen. Angenommen ein Nutzer fährt mit einem batteriebetriebenen Fahrzeug nach Hause und er will nach dem Ankommen noch den Einkauf erledigen. Da die Reichweite nicht mehr ausreicht, steckt er das Fahrzeug an die Steckdose und bekommt die Info, dass der Vorgang der Ladung 30 Minuten in Anspruch nimmt. Nach 30 Minuten kommt der Nutzer wieder und will mit dem Fahrzeug wegfahren, jedoch wurde die angezeigte Zeit korrigiert und beträgt immer noch 20 Minuten. Solche Vorfälle haben negative Auswirkungen auf die Kundenzufriedenheit und sollten im Regelfall nicht vorkommen. Eine genaue und präzise Angabe der benötigten Ladezeit ist somit essentiell.
Diese, dem Nutzer vorhergesagte Ladezeit zum Zeitpunkt des Einstecken des Ladekabels, wird bisher in Hybrid und Elektrofahrzeugen des Projektpartners Daimler AG basierend auf Berechnungen physikalischer Größen wie Temperatur, Spannung und Ladezustand erstellt. Dieses Modell weist starke Abweichungen der Vorhersage im Vergleich zur tatsächlichen Batterieladezeit auf, wie in Abbildung 1 abgebildet. Laut Untersuchungen seitens Daimler AG, ist die Vorhersage der Ladezeit in 83% der Fälle nicht zufriedenstellend.

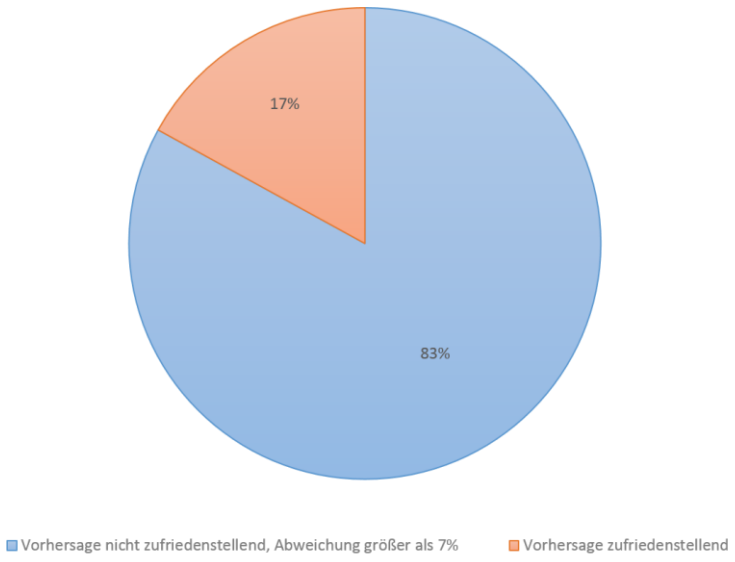

Abb. 16: Überblick Vorhersage Batterieladezeit

Im Zuge dieser Arbeit soll untersucht werden ob mittels einem datenbasierten Lernmodell diese Restladezeit der Batterie vorhergesagt werden kann. Anschließend soll geprüft werden ob dieses maschinelle Lernmodell eine Verbesserung der Vorhersage der Batterieladezeit im Vergleich zum bisherigen, bereits implementierten Modell bewirkt. Dazu sollen die bereitgestellten Daten aufbereitet, ein maschinelles Lernmodell ausgewählt und trainiert

werden. Anschließend soll die Genauigkeit des datenbasierten Modelles und des physikalischen Modelles verglichen und evaluiert werden. Abschließend sollen die Ergebnisse präsentiert, Handlungsempfehlungen und ein Ausblick gegeben werden.

4.1 Batteriebetriebene Fahrzeuge

Die aktuelle Ladeprognose der batteriebetriebenen Fahrzeuge des Projektpartners Daimler AG beruht auf einem Matlab Simulink Modell, das sich über Jahre an ein sehr komplexes Modell entwickelt hat und selbst von Mitarbeitern mit jahrelanger Erfahrung nicht in einfachen Sätzen beschrieben werden kann. Dies ist der Grund warum das Modell in folgendem Absatz nur grundlegend beschrieben wird. Aufgrund von Messgerätefehler, Wirkungsgradänderung des Gleichspannungswandlers oder des Ladegerätes, Kapazitätsänderungen der Batterie durch Alterung, verschiedener Lastzustände der Verbraucher sowie unterschiedlichem Verhalten der Batterie bei Temperaturschwankungen kommt es zu starken Abweichungen in der Prognose der Restladezeit der Batterie.

4.1.1 Ladevorgang

In Kapitel 2.1 wurde das Laden prinzipiell erläutert. In den untersuchten Ladevorgängen von Hybridfahrzeugen wird immer mit konstantem Strom geladen, wie in Abb. 1: in Bereich 1 gezeigt. Der Hintergrund dazu ist, dass erstens die Batterie vor Überspannung geschützt wird und zweitens steht bei einem Hybridfahrzeug im Notfall der Verbrennungsmotor als Alternativantrieb zur Verfügung. Es ist daher die Reichweite und der Betrieb der Batterie bis an die Leistungsgrenze weniger relevant als die Langlebigkeit der Batterie.

4.1.2 Physikalische Ladeprognose

Wie in Kapitel 2.3 erläutert, gibt es grundlegend zwei Methoden, die Restladezeit der Batterie zu bestimmen. Hierbei wird zuerst die maximal mögliche Leistung der Batterie anhand eines Leistungskennfeldes bestimmt. Dieses Leistungskennfeld ist von der Temperatur und vom Ladezustand abhängig. Anschließend wird anhand des aktuellen Ladezustandes, der aktuellen Spannung der Batteriezellen und der Gesamtbatterie und der aktuellen Kapazität der Batterie der nötige Energieinhalt zur Vollladung der Batterie berechnet. Im Folgenden wird der aktuelle Ladestrom erfasst und mittels des berechneten Energieinhaltes die Restladezeit der Batterie berechnet.

4.2 Daten

Im folgenden Kapitel wird allgemein der Aufbau der Daten beschrieben, die Datenqualität sowie auf die Datenerfassung eingegangen. Je weiter das Unterkapitel fortschreitet, desto tiefer ins Detail wird der Aufbau beschrieben.

4.2.1 Allgemeines

Die bereitgestellten Daten wurden seit Mai 2017 bis Dezember 2018 aufgezeichnet und repräsentieren 16 verschiedene Fahrzeuge von zwei verschiedenen Fahrzeugmodellen der E-Klasse Hybrid und S-Klasse Hybrid.
Die verwendeten Daten entstehen bei einem Dauerlauftest und werden von zusätzlich eingebauten Datenloggern aufgezeichnet. Ziel eines Dauerlauftestes ist es, ein Lastkollektiv abzubilden das möglichst der Realität gleicht, jedoch in einem gestauchten Zeitbereich. Ein Dauerlauftest besteht aus mehreren Teilbereichen, in denen versucht wird, alle Komponenten des Fahrzeuges auf Haltbarkeit und Verschleiß zu überprüfen. Hierbei werden unterschiedlichste Fahrbahnen und verschiedene Geschwindigkeitsbereiche abgefahren.
Die verwendeten Daten beziehen sich ausschließlich auf Hybridfahrzeuge der Marke Mercedes des Projektpartners Daimler AG. Die verbaute Kapazität und das Modell der Batterie sind in allen Tests einheitlich. Um einen Dauerlauftest realistisch durchführen zu können, muss zwischen den Tests sowie am Ende des Testes die Batterie wieder aufgeladen werden. Diese Ladevorgänge erfolgen unter den verschiedensten Bedingungen. Beispielsweise variieren die noch enthaltene Restkapazität der Batterie, die Außentemperatur, die Leistung des Ladegerätes, sowie der Zustand der Heizung oder der Klimaanlage.
Der ganze Zeitbereich eines solchen Tests wird mittels Datenloggern aufgezeichnet und gespeichert. Pro Dauerlauftest entsteht somit eine Datei, in der der ganze Dauerlauftest des Fahrens und Ladens gespeichert ist.
Diese Daten werden im Format „.atfx“ gespeichert. Dies ist ein Standard Format der “Association for Standardization of Automation and Measuring Systems” (Abkürzung: ASAM) für Daten Management und Datenanalyse [6]. ASAM ist ein Verein der Automobilindustrie mit dem Zweck der internationalen Standardisierung um durchgehenden Datenaustausch zu ermöglichen. ATFX ist ein XML basiertes Datenformat, das 2003 von ASAM eingeführt wurde. Um diese Daten in einem Programm wie Python einzulesen, wird ein anderes Format benötigt. Daher ist eine Umwandlung des vorhandenen Datenformates nötig. Ein ASAM Format das in den vorher genannten Programmen einlesbar ist, nennt sich „Measurement Data Format“ (Abkürzung: MDF) und ist seit 2009 ein eingeführter Standard der Automobilindustrie [7]. Dies ist ein binäres Datenformat und ermöglicht ein effizientes und schnelles Speichern und Bearbeiten von großen Datenmengen.
Für den in Kapitel 4 angeführten Anwendungsfall sind nur die Teilbereiche in denen die Batterie geladen wird wichtig und interessant. Um Fehler und mögliche Ursachen der auftretenden Fehler leichter zu detektieren, ist der Zeitbereich der Aufzeichnung größer als der reine Ladevorgang. Für den betrachteten Anwendungsfall sind nur Daten wichtig, in denen das Ladegerät aktiv ist und tatsächlich Strom vom externen Netz in die Batterie fließt. Dies führt dazu, dass der Ladevorgang zuerst noch auf diesen Zeitbereich zugeschnitten werden muss.

4.2.2 Datenerfassung

Wie in Abschnitt 4.2.1 erwähnt, werden alle Daten mittels verbautem Datenlogger aufgezeichnet. Es werden hierbei nicht alle im Fahrzeug auftretende Signale aufgezeichnet,

sondern nur ausgewählten Signale der verschiedenen Bussysteme. Je nach verbautem Datenlogger im Fahrzeug und je nach Konfiguration dessen, können die aufgezeichneten Signale variieren. Das Grundgerüst der Daten ist somit nicht einheitlich. Diese Messdaten werden mit der Abtastfrequenz von 1 Hz aufgezeichnet. Prinzipiell kann der im Fahrzeug verbaute Datenlogger Full-Trace und signalbasiert aufzeichnen. Im Full-Trace Betrieb wird jedes Signal exakt in der Frequenz aufgezeichnet in der es gesendet wird. Bei signalbasierter Aufzeichnung, muss zuerst eine Abtastrate (Samplingrate) und eine Samplingtiefe festgelegt werden [49]. Hierbei wird ein zeitkontinuierlichen Signales in ein Zeit und amplitudendiskretes Signal umgewandelt. Die Abtastrate ist die Häufigkeit der Messung eines zeitkontinuierlichen Signales und die Samplingtiefe gibt die Anzahl der Bits pro Abtastung an. Als Beispiel wird das Signal „Ladezustand_real“ in Abb. 17: betrachtet. Hier haben wir eine Abtastfrequenz von 1Hz und eine Samplingtiefe von 10 Bit. Diese Samplingtiefe führt dazu, dass in unserem Beispiel $2^{10} = 1024\, m\ddot{o}gliche\, Zust\ddot{a}nde$ abgebildet werden können. Dies führt zu einer Signalauflösung von $\frac{1}{2^{10}} = \text{ca. } 0{,}1\%$.

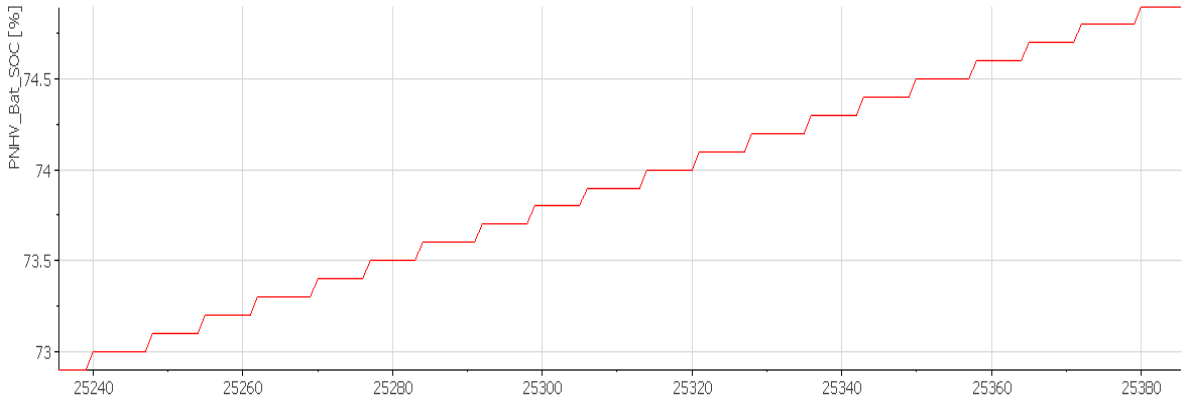

Abb. 17: Samplingtiefe: Grund des Sprungverhaltens

Wie in Abb. 17: ersichtlich, führt dies zu den sichtbaren Sprüngen, da nur eine Änderung von 0,1% aufgezeichnet wird.

4.2.3 Datengrundlage

Seitens Daimler wurden zwei Dateiformate zur Verfügung gestellt, MDF und CSV. Um eine MDF Datei in Python einzulesen, wird ein zusätzliches Paket benötigt mit dem Namen „asammdf“ [49]. Zuerst wurde das Dateiformat MDF getestet, da dies viel weniger Speicher benötigt als eine CSV Datei. Das Einlesen des MDF Formates in Python war nicht zufriedenstellend, da von 100 Dateien mehr als 20 Dateien aufgrund eines Skript Fehlers nicht einlesbar waren. Aus diesem Grund wurde das Format CSV gewählt. Das Datenformat CSV war ohne Fehler in Python möglich.
Die Dateien wurden vor dem Einlesen so aufbereitet, dass eine CSV Datei nur einen Ladevorgang abbildet. Diese Daten werden in folgende 3 Kategorien eingeteilt, wie in Abb. 18: dargestellt:

- Die Ladung war in Ordnung und die Batterie wurde vollgeladen. Dies sind 3131 Ladevorgänge und farblich in grün markiert.
- Die Ladung war in Ordnung, jedoch unvollständig. Es fand keine Vollladung der Batterie statt. Dies sind 1465 Ladevorgänge und sind in grau markiert
- Die Ladung war nicht erfolgreich. Es gab einen oder mehrere Fehler während des Ladevorganges. Dies sind 2690 Ladevorgänge und farblich in Rot markiert.

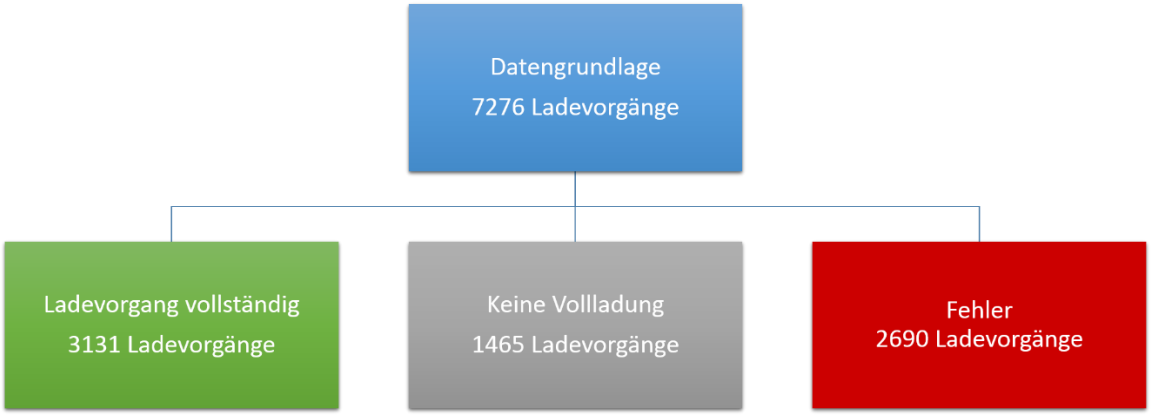

Abb. 18: Datengrundlage mit Kategorisierung

Die Summe aller von Daimler zur Verfügung gestellten Daten umfasst 7276 Ladevorgänge. Relevant sind jedoch nur diese, in denen der Ladevorgang physikalisch korrekt abgelaufen ist. Daher werden fehlerhafte Dateien, in Abb. 18: als rot markiert, nicht miteingeschlossen. Das Ziel ist es, die Vorhersage der Restladezeit der Batterie bis zu einer Vollladung von 100% zu verbessern, daher werden die Ladevorgänge, bei denen keine Vollladung stattfand, nicht miteingeschlossen. Übrig bleiben somit 3131 verwendbare Ladevorgänge, bei denen der physikalische Ladevorgang erfolgreich und fehlerfrei verlaufen ist. Untersucht man nun diese 3131 Ladevorgänge hinsichtlich der Prognose der Restladezeit der Batterie, wird ersichtlich, dass nur 17% der Prognosen das Bewertungskriterium von $\pm 7\%$ erfüllen. Dies ist in Abb. 19: abgebildet. Ziel ist es, ein datenbasiertes Modell zu bilden und zu validieren, das die restlichen 83% der Ladevorgänge besser vorhersagen kann, als das bisherige physikalische Modell.

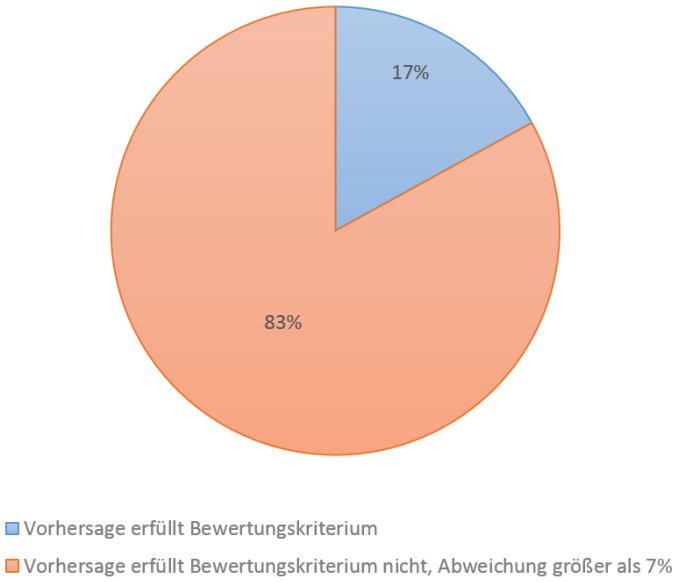

Abb. 19: Prognosegüte des bisherigen physikalischen Modelles

4.2.4 Dateiaufbau

Eine Datei ist aus Spalten und Zeilen aufgebaut und bildet dabei eine Matrix. Es ist somit eine Matrix an Eingangsvariablen, die aus Zahlen sowie auch Wörter bestehen. Die Spalten beinhalten die verschiedenen Signale, auch Features genannt, und die Zeilen die verschiedenen Abtastungszeitpunkte. Wie in Kapitel 4.2.2 beschrieben ist die Abtastfrequenz 1Hz und somit wird jedes im Datenlogger eingestellte Signal einmal pro Sekunde aufgezeichnet.
Wie in Abschnitt 4.2.2 erwähnt, können je nach verbautem Datenlogger im Fahrzeug und je nach Konfiguration dessen, die beinhalteten Signale, also die Anzahl der Spalten der Matrix, variieren. Der Inhalt der Daten und die aufgezeichneten Signale sind somit nicht einheitlich.
Die Anzahl der Zeilen ist unterschiedlich, da es sich hier um Zeitreihen handelt. Je nach Dauer des in der Datei gespeicherten Ladevorganges variiert diese Zeitdauer.

4.3 Datenanalyse und Bearbeitung

Im Normalfall entsprechen Messdaten, ohne vorherige Bearbeitung nicht dem nötigen Format um mit maschinellem Lernen ein gutes Ergebnis zu erzielen. Daher ist die Datenaufbereitung ein Grundelement in jeder Machine Learning Anwendung. Für eine optimale Leistung müssen leere Zeilen vermieden, Ausreißer detektiert und die Signalanzahl minimiert werden.
Viele Machine Learning Algorithmen liefern bessere Ergebnisse wenn man die Daten skaliert in dem man den Datensatz zum Beispiel in einen Bereich von [0,1] transformiert [11]. In diesem Kapitel wird erläutert, wie die Daten bearbeitet wurden.

Weiters ist es sinnvoll, die Eingangssignale zu analysieren und Signale (im Weiteren als Features bezeichnet) mit hoher Korrelation zu eliminieren um weniger Speicher zu benötigen und um den Algorithmus zu beschleunigen.
Am Ende werden die Daten zufällig in einen Trainingsanteil und Testanteil aufgeteilt. Mit dem Trainingsdatensatz werden die Gewichtungen des Algorithmus trainiert und gebildet und mit dem Testdatensatz anschließend evaluiert.

4.3.1 Datenverständnis

Im aktuellen Ladesystem gibt es drei verschiedene Kenngrößen des Ladezustandes der Batterie. Das erste Signal wird als „Ladezustand_real“ bezeichnet und spiegelt den echten Ladezustand wieder. Wie aus Abb. 20: ersichtlich, wird ein echter Ladezustand von knapp 90% erreicht. Das zweite Signal wird als „Ladezustand_Display“ bezeichnet und spiegelt den angezeigten Ladezustand am Fahrzeugdisplay wieder. Ersichtlich ist, dass der Kunde am Display bereits 100% angezeigt bekommt obwohl der echte Ladezustand, der mit einer grünen Linie wiedergespiegelt wird, noch steigt. Das dritte Signal wird als „Ladezustand_Steuergerät“ bezeichnet und spiegelt die interne Rechengröße des Steuergerätes wieder. Diese interne Rechengröße beinhaltet weniger sprunghaftes Verhalten und wird für die Regelung des Steuergerätes verwendet.

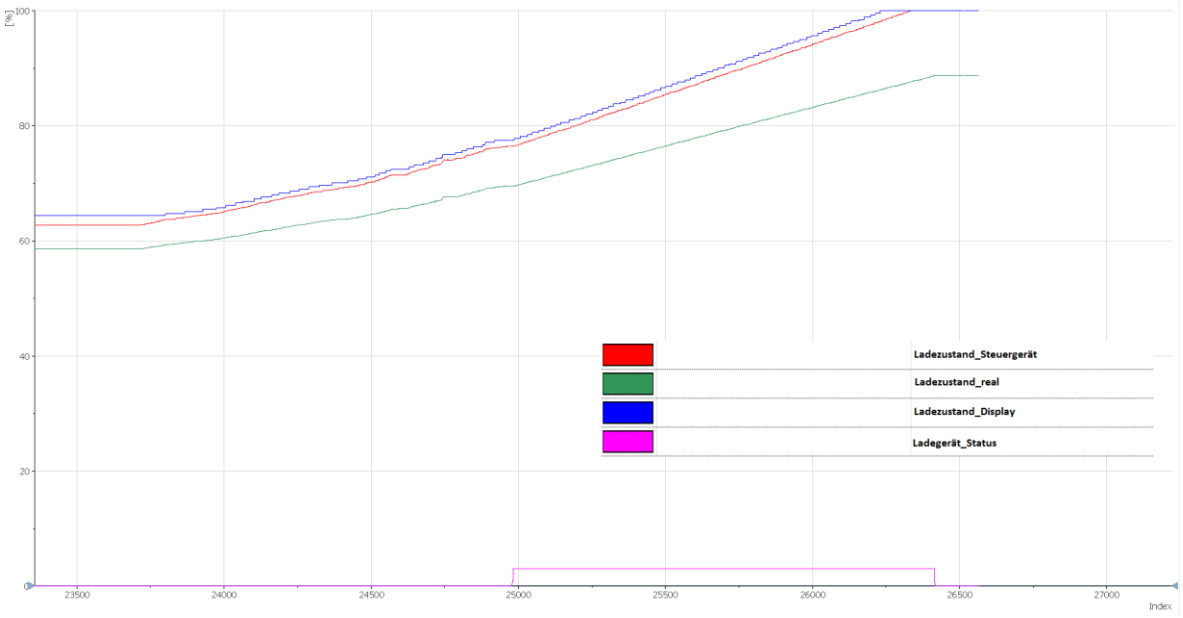

Abb. 20: Ladezustand (State of Charge)

Die letzte ersichtliche Kenngröße wird mit „Ladegerät_Status“ bezeichnet und spiegelt den Status des internen Ladegerätes wieder. Der Status 0 kennzeichnet den Niedrigenergie Modus, Status 1 den Standby Modus und Status 2 den Modus „Ladebereit“. Aufgrund der großen Auflösung ist der Zustand 2 in Abb. 20: nicht ersichtlich. Zustand 3 spiegelt den aktiven Lademodus wieder und ist deutlich anhand des in Abb. 20: ersichtlichen Sprunges erkennbar. Befindet sich dieses Signal im Zustand 3, wird die Batterie geladen. Die Daten müssen somit

auf den Zeitbereich zugeschnitten werden, in denen sich das Signal „Ladegerät_Status“ im Zustand 3 befindet.

Im Folgenden werden die Möglichkeiten beschrieben die zu einem Abbruch des Ladevorganges führen. Es gibt 5 Möglichkeiten die zum Abbruch einer Ladung führen:

1. Es wurde die maximale zulässige Energie in die Batterie geladen. Dazu berechnet das Steuergerät immer die aktuelle Energie, die in die Batterie geladen wurde und beendet den Ladevorgang wenn ein vom Batteriesystem definierter Grenzwert erreicht wird. Bei Plug-In Hybrid Fahrzeugen liegt dies ca. bei 10 kWh.
2. Der reale Ladezustand erreicht einen definierten Grenzwert. Bei Plug-In Hybrid Fahrzeugen liegt dieser Grenzwert bei ca. 90%.
3. Der Batteriestrom sinkt unter einen gewissen Wert für eine bestimmte Zeit. Der Grenzwert liegt bei ca. 1A für mehr als 5 Sekunden.
4. Ein durch den Nutzer voreingestellter Ladezustand wird erreicht oder der Nutzer bricht den Vorgang durch Aufsperren des Fahrzeuges und anschließendem Wegfahren ab.
5. Es wird ein Fehler im Ladegerät oder in der Ladeinfrastruktur identifiziert.

Die ersten 3 Fälle führen zu einer Vollladung der Batterie. Im 4. Fall wird ein Ladezustand vom Nutzer vorgegeben und bis zu diesem Wert wird die Batterie geladen. Dies erklärt die in Abb. 18: gezeigte Aufteilung. Kommt es zu einer Vollladung durch die ersten 3 Möglichkeiten, wird die Batterie vollgeladen und zählt zu den 3131 Ladevorgängen, die farblich in Grün markiert sind. Im 4. Fall wird die Batterie nicht vollgeladen und daher werden diese Daten für diesen Anwendungsfall nicht berücksichtigt. Diese Ladevorgänge werden in Abb. 18: als grau in die Kategorie „Keine Vollladung“ eingeteilt.
Im 5. Fall tritt ein Fehler auf und dieser Ladevorgang wird somit der Kategorie „Fehler“ in Abb. 18: zugewiesen.

4.3.2 Aufbereitung

Wie in Kapitel 4.2.2 erwähnt, werden in den Daten jeweils unterschiedliche Signale aufgezeichnet, die teilweise unvollständig sind. Ein maschineller Lernalgorithmus benötigt lückenfreie Daten mit einheitlichen Features. In diesem Abschnitt wird das Vorgehen beschrieben um solche Daten zu erhalten.

Features entfernen

Alle Features die nur 0 oder NaN (engl.: Not a Number, nicht darstellbarer Wert) Werte enthalten wurden beim Einlesen in Python gelöscht, da in diesen Features keine Information enthalten ist.

Interpolation

Um fehlende Werte in einem Feature zu ersetzen wurden alle fehlenden Werte linear interpoliert wie in Abb. 21: dargestellt und in Gleichung 11 berechnet.

$$y_u = y_1 + \frac{x_u - x_1}{x_2 - x_1} * (y_2 - y_1) \tag{11}$$

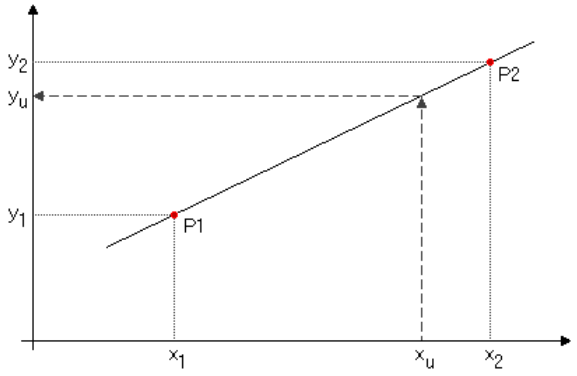

Abb. 21: Lineare Interpolation

Filtern des Ladevorganges

In jeder Datei wird, wie in Kapitel 4.2.2 beschrieben, ein größerer Zeitbereich als der Ladezeitraum aufgezeichnet. In diesem Anwendungsfall wird nur der Bereich des Ladens benötigt und daher muss jede Datei auf den Zeitbereich zugeschnitten werden, in dem das Ladegerät aktiv ist. Dieser trifft zu, wie in Kapitel 4.3.1 beschrieben, wenn das Signal „Ladegerät_Status“ den Zustand 3 einnimmt.

Berechnung der Restladezeit

Die Restladezeit wird, wie in Gleichung 12 beschrieben, pro Zeitpunkt berechnet.

$$Restladezeit = t_{Ende} - t \tag{12}$$

Wie in Kapitel 4.2.2 beschrieben, werden alle Signale sekündlich aufgezeichnet, somit ergibt sich für die Berechnung der Restladezeit eine Gerade, bespielhaft in Abb. 22: dargestellt.

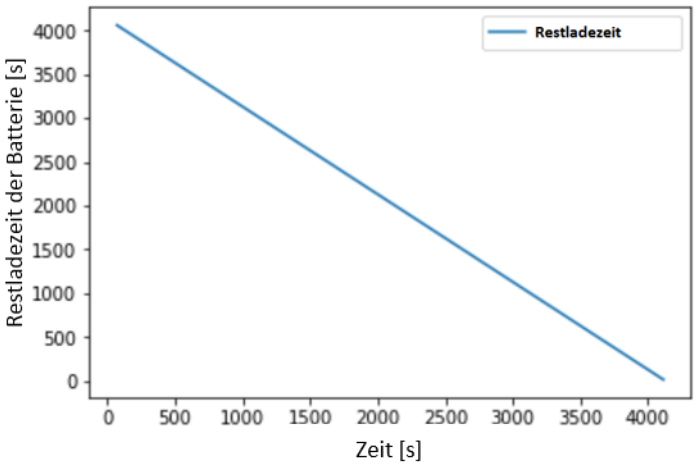

Abb. 22: Restladezeit

Auf wesentliche Signale beschränken

Im Datenlogger werden rund 500 Signale aufgezeichnet, diese sind jedoch nicht alle für den Ladevorgang relevant. Signale wie Gaspedalstellung, Lenkradwinkel oder der Radschlupf spielen während dem Laden im Stillstand keine Rolle.
In Absprache mit den Ladeexperten des Projektpartners Daimler AG wurden folgende Signale für die weitere Analyse ausgewählt: Signale der Hochvolt Batterie, der Niedrigvolt Batterie, des Ladegerätes, der Gleichstromwandler, der Heizung, der Klimaanlage, der Zeitmessung, des Kilometerstandes, der laderelevanten Fehlerbits, sowie alle Temperaturen der vorher genannten Bauteile verwendet. Im Anhang ist eine genaue Aufschlüsselung der ausgewählten Signale abgebildet.

4.3.3 Ansätze um Zeitreihen in ein maschinelles Lernproblem überzuführen

Untersucht werden 3 verschiedene Ansätze, um Zeitreihen in ein maschinelles Lernproblem überzuführen. Durch diese 3 Ansätze sollen die Daten in ein Datenformat, wie in Kapitel 3.3.2 beschrieben, transformiert werden.

4.3.3.1 Ansatz 1: Zeitpunktbetrachtung

In diesem Ansatz wird zu jedem Zeitpunkt ein Zustandsbild aller Signale gespeichert als Eingangsgröße. Die Ausgangsgröße ist die Restladezeit, die zu diesem Zeitpunkt benötigt wird, um die Batterie auf 100% zu laden. Beispielhaft wird dies anhand eines Signales, der Batteriespannung, in Abb. 23: gezeigt. Die Zeitschritte betragen jeweils ein Sekunde.

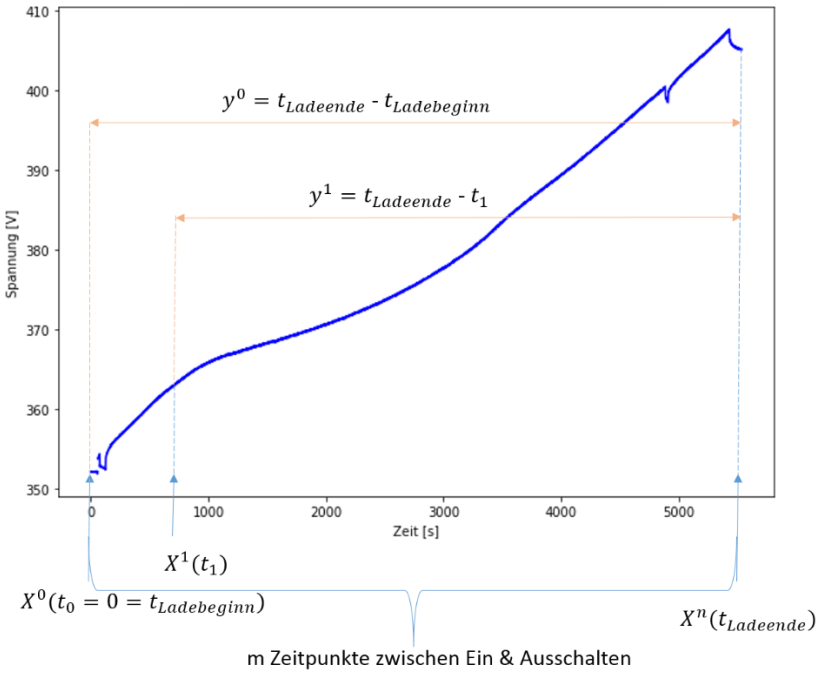

Abb. 23: Ansatz 1 - Zeitpunktbetrachtung

Mathematisch wird dies durch den Vektor $X^0 = (x_1^0, x_2^0, x_3^0, \dots x_k^0)$ beschrieben, wobei $x_1^0, x_2^0, x_3^0, \dots x_k^0$ der jeweilige Zustand der Signale $x_1 \dots x_k$ zum Zeitpunkt 0 ist. Anschließend wird dies auf alle m Zeitpunkte angewendet. Somit wird ein Datenformat, wie in Kapitel 3.3.2 beschrieben, erreicht.
Es entsteht ein Datensatz wie in Gleichung 13 gezeigt, mit dem Vektor X^0als Eingangsgröße und der Restladezeit y^0 als Ausgangsgröße. Der Vorteil dieser Methode besteht in der Einfachheit und darin, eine maximale Anzahl an Samples zu erreichen.

$$\begin{pmatrix} X^0 = (x_1^0, x_2^0, x_3^0, \dots x_k^0) & y^0 \\ \vdots & \vdots \\ X^n = (x_1^n, x_2^n, x_3^n, \dots x_k^n) & y^n \end{pmatrix} \qquad \text{Gleichung (13)}$$

4.3.3.2 Ansatz 2: Zeitfenster

In diesem Ansatz werden mehrere Zeitfenster aus jedem Ladevorgang herausgeschnitten und als Eingangsgröße im Sample gespeichert. Die Ausgangsvariable ist die Restladezeit bis zur Vollladung der Batterie. Hierbei wurden 15 Sekunden als optimaler Bereich des Zeitfensters gewählt, weil dies die minimale Zeitdauer ist, um eine Veränderung der Signale zu erkennen. Es wird ein Zeitfenster mit der Größe von 15 Sekunden herausgeschnitten und im Datensatz gespeichert. Danach folgt eine Pause von 15 Sekunden und das zweite Zeitfenster von 15 Sekunden wird herausgeschnitten. Dies wird bis zum letzten Zeitfenster fortgesetzt, an dem die Restladezeit 0 ist und der Ladezustand der Batterie 100% beträgt. Die Anzahl der

Zeitfenster hängt von der Zeitdauer des Ladevorganges ab. In Abb. 24: ist dies am Beispiel eines Signales, der Batteriespannung, gezeigt.

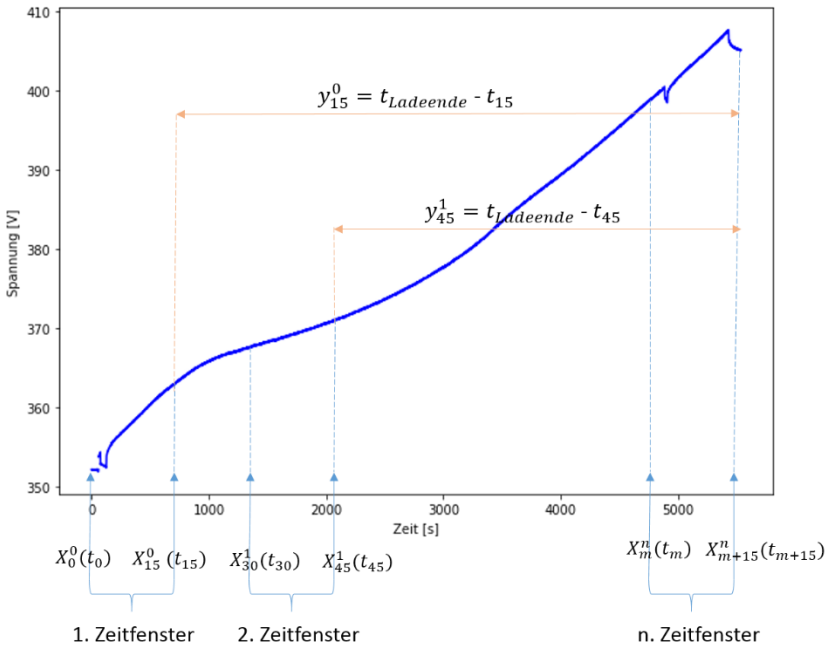

Abb. 24: Ansatz 2

In diesem Ansatz werden pro Sample zwei Zustandsschaubilder gespeichert, jeweils um 15 Sekunden versetzt. Somit ist eine Veränderung der einzelnen Eingangsgrößen in jedem Sample abgebildet.
Mathematisch wird dies durch den Vektor $\mathrm{X}_0^0 = (x_1^0, x_2^0, x_3^0, \dots x_k^0)$, wobei $x_1^0, x_2^0, x_3^0, \dots x_k^0$ der jeweilige Zustand der Signale $x_1 \dots x_k$ zum Zeitpunkt 0 ist, beschrieben. Der Vektor $\mathrm{X}_{15}^0 = (x_1^{15}, x_2^{15}, x_3^{15}, \dots x_k^{15})$ bildet dabei den Zustand der Signale zum Zeitpunkt 15 ab. Anschließend wird dies auf alle $0{,}30{,}60 \dots m$ Zeitpunkte angewendet. Somit ergibt sich ein Datenformat, wie in Abb. 9: beschrieben. Hierbei sind die Eingangsgrößen als X und die Ausgangsgröße als y abgebildet.

$$\begin{pmatrix} X_0^0 & X_{15}^0 & ,y_{15}^0 \\ \vdots & \vdots & \vdots \\ X_m^n & X_{m+15}^n & ,y_{m+15}^n \end{pmatrix}$$

Gleichung (14)

4.3.3.3 Ansatz 3: Zeitfenster mit beschreibender Statistik

In diesem Ansatz wird zusätzlich zu den zwei Zustandsschaubildern noch eine Information über den Verlauf der Signale innerhalb des Zeitfensters gespeichert wird, wie in Abb. 25: schematisch gezeigt. Diese Statistik setzt sich aus dem Mittelwert, der Varianz, der Standardabweichung, dem Minimum und dem Maximum folgender Signale zusammen:

- Batteriespannung

- Batterieladezustand
- Batteriestrom

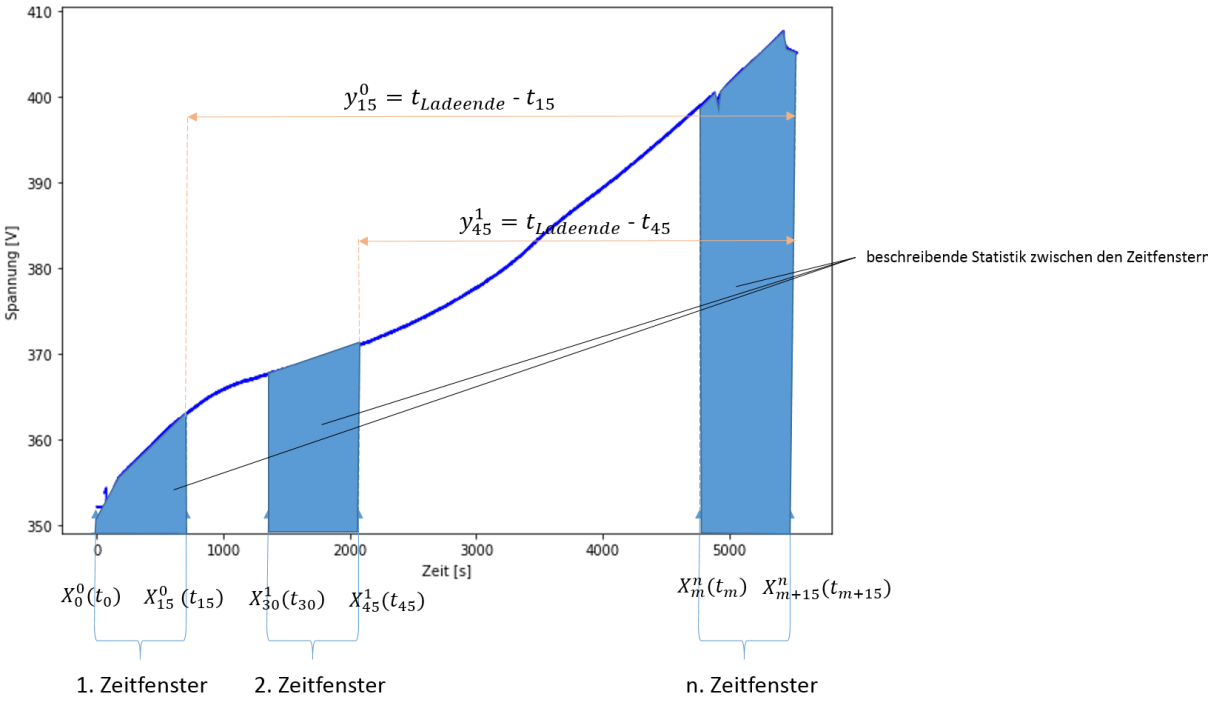

Abb. 25: Ansatz 3

Diese drei Signale wurden ausgewählt, weil dies die drei wichtigsten Features sind, die eine Aussage über den Zustand der Batterie liefern.
Die Basis bildet, gleich wie in Ansatz 2 ein Zeitfenster von 15 Sekunden. Zusätzlich kommt anschließend noch die Differenz der beiden Vektoren $X_{\mathrm{m}}^{\mathrm{n}}$ und $X_{\mathrm{m+15}}^{\mathrm{n}}$ sowie eine beschreibende Statistik zwischen den Zeitpunkten. Dies wird in Gleichung 15 mittels $\mathrm{statistik}(\mathrm{X_m^n : X_{m+15}^n})$ beschrieben. Die Zeitpunkte bleiben gleich mit $0{,}30{,}60 \ldots m$. Somit ergibt sich ein Datenformat, wie in Abb. 9: beschrieben.

$$\begin{pmatrix} \mathrm{X_0^0} & \mathrm{X_{15}^0} & \mathrm{X_{15}^0 - X_0^0} & \mathrm{statistik(X_0^0 : X_{15}^0),} & \mathrm{y_{15}^0} \\ \vdots & \vdots & & \vdots & \\ \mathrm{X_m^n} & \mathrm{X_{m+15}^n} & \mathrm{X_m^n - X_{m+15}^n} & \mathrm{statistik(X_m^n : X_{m+15}^n),} & \mathrm{y_{m+15}^n} \end{pmatrix} \qquad \text{Gleichung (15)}$$

4.3.4 Auswahl der eingelesenen Ladevorgänge

Wie in Kapitel 4.3.2 erwähnt, ist die Aufzeichnung der Signale je nach Datenlogger Konfiguration unterschiedlich. Wichtige Signale, wie die tatsächliche momentane Leistung des Ladegerätes, die Stromaufnahme der Batterie, Batteriespannung und die Anzahl der verwendeten Phasen sind Kernsignale, die in jedem Ladevorgang aufgezeichnet werden müssen. Bezogen auf die Datengrundlage von 1. Dezember 2018, wie in Abschnitt 4.3.7

erläutert, sind in 822 Ladevorgängen die vorher genannten Signale nicht vorhanden. Diese Ladevorgänge wurden in den ersten Iterationen trotzdem berücksichtigt, um die Datenanzahl möglichst groß zu halten. Ab dem Iterationsschritt 8 der Modellbildung wurden diese 822 Ladevorgänge entfernt und daher weiter nicht mehr berücksichtigt.

4.3.5 Unterteilung abhängig der Ladeleistung

In Abb. 26: ist die Verteilung der Ladeleistung, anhand eines Histogrammes, der Ladevorgänge abgebildet. Ersichtlich ist eine Häufigkeitsverteilung der Ladevorgänge, wobei Leistungen über 5 kW eine Häufigkeit von ca. 85% aufweisen. Leistungen unter 5kW weisen eine geringe Häufigkeit auf. Aufgrund der unterschiedlichen Repräsentativität der Ladeleistung, wurden die Daten anhand der Ladeleistung in Leistungen über 5kW und Leistungen unter 5kW aufgeteilt.

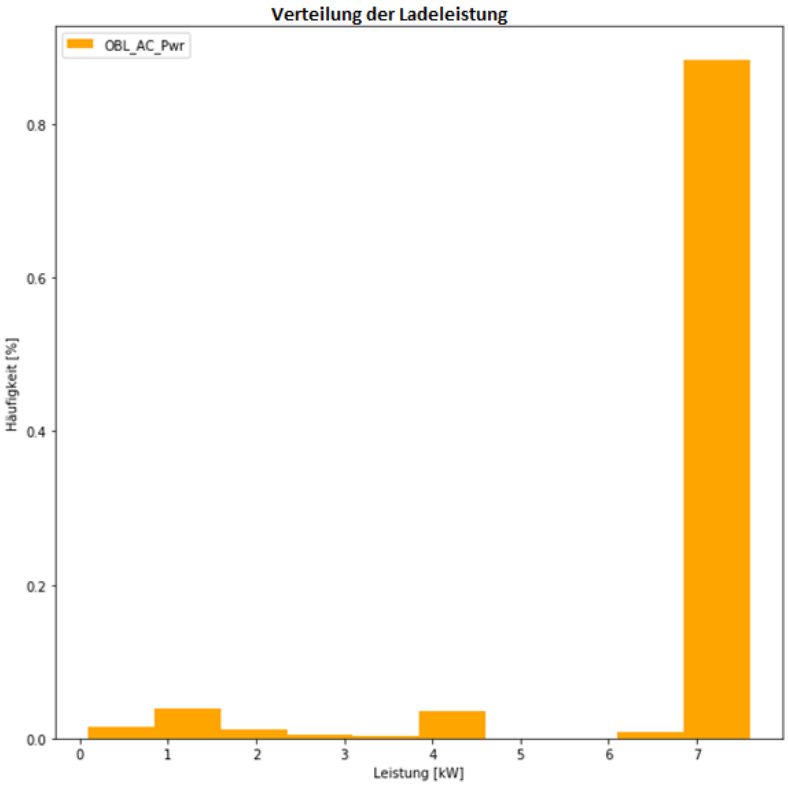

Abb. 26: Histogramm: Ladeleistung

4.3.6 Skalierung der Daten

Die Daten wurden mittels „Standardisierung“, wie in Kapitel 3.3.2.3 beschrieben, skaliert. Hierbei wird eine Standard Normalverteilung erreicht, indem der Mittelwert um den Wert 0 zentriert wird und die Standardabweichung bei 1 liegt.

4.3.7 Datenverfügbarkeit

Die Datenbereitstellung seitens Daimler AG begann im September 2018 mit einem Ladevorgang und mit fortschreitender Zeit wurden immer mehr Daten bereitgestellt. Ein zeitlicher Verlauf dieses Vorganges ist in Abb. 27: gezeigt. Hierbei werden nur die Ladevorgänge betrachtet, die in Abb. 18: als vollständige Ladung kategorisiert wurden.

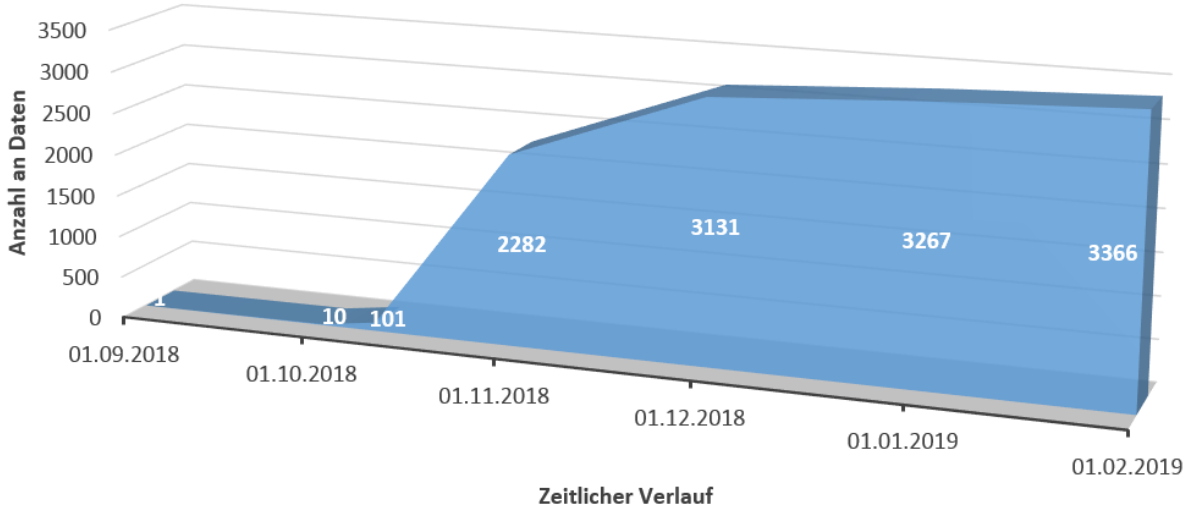

Abb. 27: Datenverfügbarkeit

Um ein datenbasiertes Modell zu bilden ist eine große Datenbasis essentiell. Anfang November wurde eine Anzahl an 101 Ladevorgängen bereitgestellt. Dies ist der Grund dafür, dass die Modellbildung zuerst mit 101 Ladevorgängen begonnen hat. Nach der Datenbereitstellung wurden anschließend stückweise Daten hinzugefügt. Dies hat die Masterarbeit zeitlich nach hinten versetzt, da gewisse Zusammenhänge in den Daten erst später erkannt wurden, als eine höhere Anzahl an Daten zur Verfügung stand.

4.4 Machine Learning

4.4.1 Modellübersicht

Der im Kapitel 4 beschriebene Anwendungsfall ist ein klarer Fall von überwachtem Lernen. Es liegen Eingabeparameter vor, die den Output, die Restladezeit, beeinflussen. Die Aufbereitung der Daten wurde in Kapitel 4.3.2 bereits erläutert.
Grundsätzlich gibt es zwei verschiedene Arten des überwachten Lernens, Klassifikation und Regression, wie in Kapitel 3.2.1 bereits erläutert. Ziel einer Klassifikation ist es, eine Klasse vorherzusagen, wie in Kapitel 3.2.1 beschrieben. In diesem Anwendungsfall ist es das Ziel, eine kontinuierliche Größe vorherzusagen [39]. Daher handelt es sich um ein Regressionsproblem.

Grundlegendes Ziel dieser Arbeit war es, ein Modell zu wählen das im Nachhinein verständlich und erklärbar ist. Daher führte die Entscheidungswahl zuerst zu einem **linearen Lasso Modell**. Durch die Gewichtung ist es möglich, nachzuvollziehen wie eine Vorhersage berechnet wurde. Dieses Modell ist somit einfach verständlich und nachvollziehbar. Daher wird die lineare Lasso Regression als erstes Modell trainiert und validiert.
Die zweite Wahl fiel auf einen **Zufallswald**, da dieses Modell auch nichtlineare Zusammenhänge abbilden kann.
Diese beiden Modelle wurden in Kapitel 3.3.3 bereits erläutert.

4.4.2 Bewertung der Modelle

Vor der Modellbildung, werden anschließend die Bewertungskriterien erläutert, einerseits um das Modell zum bisherigen physikalischen Modell zu vergleichen und andererseits die Bewertungsmethode um eine Bewertung zwischen den Modellvarianten vorzunehmen.
Als Bewertung zwischen verschiedener RF Modellvarianten wurde das Bestimmtheitsmaß und der Root Mean Square Error verwendet, wie in Kapitel 3.3.4.1 beschrieben.

4.4.2.1 Bewertung zum bisherigen physikalischen Modell

Um einen Vergleich zum bisherigen physikalischen Modell der Prognose herzustellen, muss das Bewertungskriterium exakt gleich sein wie bisher. In Abb. 16: wurde erläutert, dass 83% der Prognosen das Bewertungskriterium von ±7% nicht erfüllen. Hierzu wurde der erste Vorhersagezeitpunkt jedes Ladevorganges betrachtet, wie in Abb. 28: abgebildet. Liegt dieser Zeitpunkt im Bereich von ±7% zur tatsächlichen Restladezeit der Batterie wird die Vorhersage als gut bewertet.

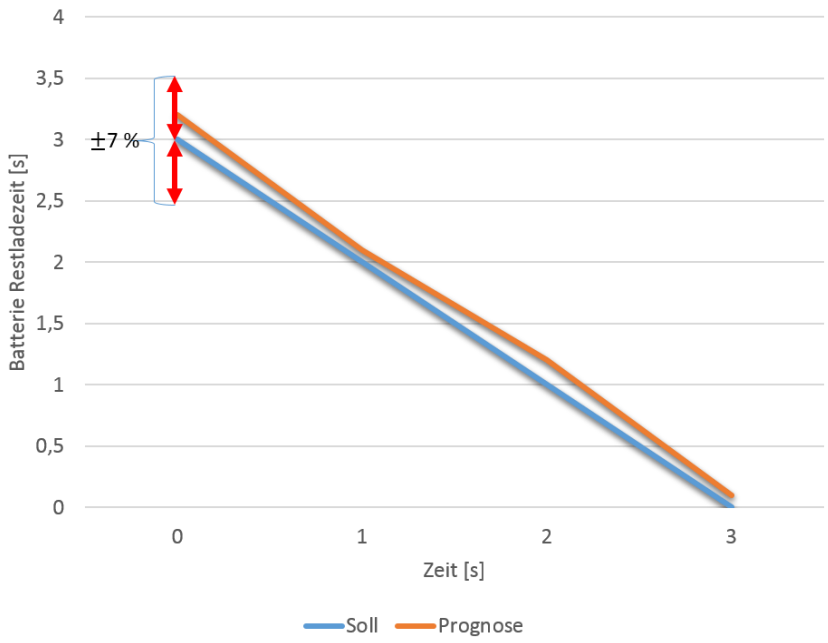

Abb. 28: Bewertung anhand der Anfangsprognose

Liegt die Vorhersage außerhalb dieses Prozentbereiches, wird sie als schlecht bewertet. Diese 7% wurden vom Projektpartner Daimler AG vorgegeben.

4.4.2.2 Bewertung über ganzen Zeitbereich

Um eine Aussage über den kompletten Zeitbereich eines Ladevorganges und dessen Prognosegüte zu bekommen, wurde ein Durchschnitt der Abweichungen berechnet. In Abb. 29: wird das prinzipiell gezeigt. Hierbei sind die Zeitpunkte 0 bis 3 betrachtet und zu jedem Zeitpunkt wird die in grau dargestellte Abweichung in Prozent berechnet, wie in Gleichung 16 und 17 beschrieben.

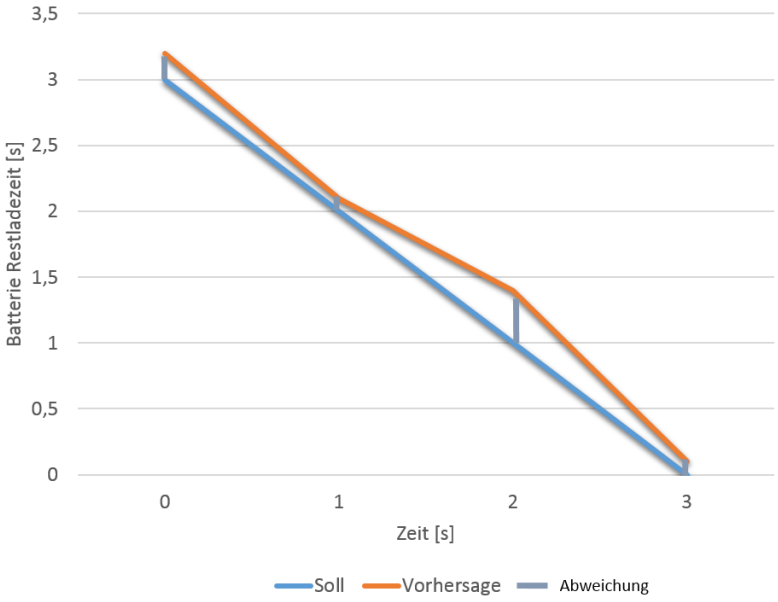

Abb. 29: Bewertung mittels der mittleren Abweichung

Anschließend wird der Betrag der Abweichungen gebildet, aufsummiert und der Durchschnitt gebildet. Liegt der Durchschnitt dieser prozentuellen Abweichung unter 7% wird der Ladevorgang als gut bewertet, wie in Gleichung 17 ersichtlich.

$$Abweichung_{\%} = \frac{Vorhersage - Soll}{Soll} * 100 \qquad (16)$$

$$\emptyset\left(\left|\sum Abweichung_{\%}\right|\right) < \quad 7\% \qquad (17)$$

4.4.3 Modellbildung

In diesem Abschnitt wird der iterative Vorgang der Modellbildung erklärt. In Abb. 30: ist die prinzipielle Vorgangsweise abgebildet. Zuerst werden die drei Ansätze, wie in 4.3.3 erläutert, mittels beider Modelle Random Forest (Abkürzung: RF) und lineare Lasso Regression

(Abkürzung: LLR) verglichen und anhand der Vorhersagegenauigkeit ausgewählt. Anschließend werden mittels dem gewählten Ansatz die beiden Modelle LLR und RF verglichen und ausgewählt. Weiters wird die Modellbildung des ausgewählten Modelles anhand einer geringen Datenanzahl gestartet. Schritt für Schritt werden mehr Daten hinzugefügt bis schließlich alle Daten verwendet werden. Weiters werden die Modelle untereinander verglichen und das Beste ausgewählt. Das Modell mit der besten Prognosegüte wird anschließend mit dem bisherigen Modell, wie in 4.1.2 erläutert, verglichen.

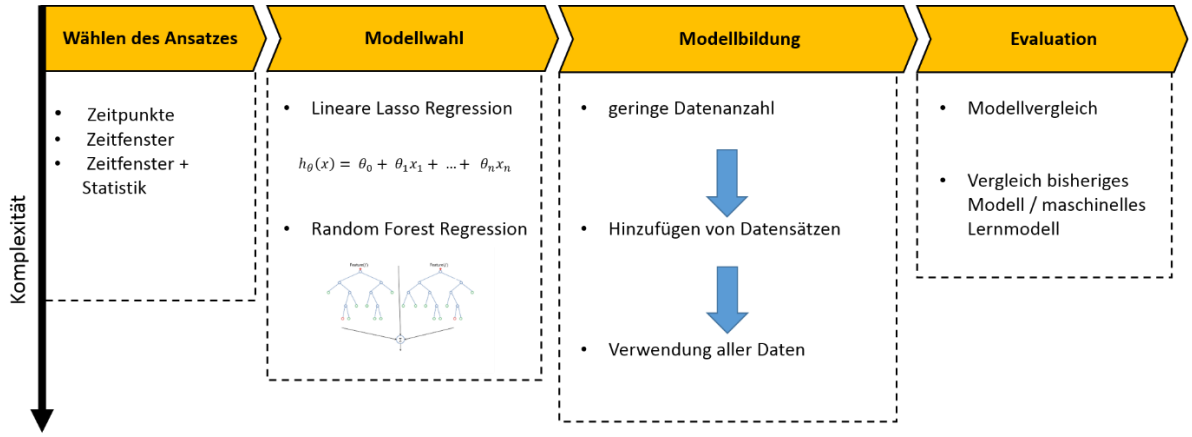

Abb. 30: Vorgehensweise Modellbildung

In Abb. 31: wird eine detaillierte Übersicht der Modellbildung gegeben. Aufgrund der Datenverfügbarkeit, wie in Kapitel 4.3.7 beschrieben, wurden die ersten zwei Modelle anhand von 101 Ladevorgängen trainiert und getestet. Im ersten Modell wurde untersucht welcher der bereits erwähnten Ansätze eine bessere Prognosegüte aufweist. Im zweiten Modell wurden die beiden Modelle Random Forest und Lineare Lasso Regression gegenübergestellt.
Das bisherige, physikalische Modell liefert, wie in Kapitel 4.2.3 beschrieben in 17% der Ladevorgänge eine gute Vorhersage der Batterieladezeit. Unter der Annahme, dass diese Daten physikalisch ähnlich sind, war es naheliegend diese zuerst zu betrachten. Aus diesem Grund wurde das dritte Modell mit diesen Daten trainiert und getestet.

Modell Nr.	Datenanzahl	Vorgehen	Ziel
1	101	Vergleich der drei Ansätze mittels RF und LLR	Bestimmen des besten Ansatzes
2	101	Vergleich RF und LLR mittels Zeitfenster Ansatz	Bestimmen des besten Modelles
3	536	RF Modellbildung anhand der bisher gut prognostizierten Daten	Eignung dieses Anwendungsfalles für maschinelles Lernen
4	3131	Training 20% , Test 80%	Anwendbarkeit Modell 3 auf restliche Datensätze
5	3131	Training 80%, Test 20%	Bestätigung der getroffenen Aussagen von Modell 4
6	3023	RF Unterteilung Ladedauer kleiner 6000s	Sample Anzahl möglichst hoch halten, Prognosegüte verbessern
7	108	RF Unterteilung Ladedauer größer 6000s	Sample Anzahl möglichst hoch halten, Prognosegüte verbessern
8	2067	RF Ladevorgänge > 5kW , reduzierte Sample Anzahl	Prognosegüte verbessern
9	242	RF Ladevorgänge < 5kW , reduzierte Sample Anzahl	Prognosegüte verbessern
10	2067	Kreuzvalidierung Modell 8	Bewertung der Modelle bzgl. des ganzen Datensatzes
11	242	Kreuzvalidierung Modell 9	Bewertung der Modelle bzgl. des ganzen Datensatzes
12	2309	RF Klassifikation	Modellunterscheidung mittels ML
13	2544	Rastersuche des RF Modelles	Parameteroptimierung

Abb. 31: detaillierte Übersicht der Modellbildung

Aufgrund der guten Prognosegüte des dritten Modelles, wurde dieses Modell anhand der restlichen, verbleibenden Daten im vierten Iterationsschritt / Modellbildungsschritt getestet. Hierbei wurde eine unkonventionelle Aufteilung der Trainings- und Testdaten von 20% zu 80% verwendet. Um die im vierten Iterationsschritt getroffenen Aussagen zu bestätigen, wurde im fünften Schritt eine konventionelle Aufteilung von 80% Trainings- und 20% Testdaten verwendet. Im fünften Schritt der Modellbildung wurden alle zur Verfügung stehenden Datensätze verwendet.
In Modell 6 und 7 wurden die Ladevorgänge anhand der Ladedauer unterteilt, mit dem Ziel die Sample Anzahl möglichst hoch zu halten. In Modell 8 und 9 wurden die Ladevorgänge anhand der Leistung unterteilt. Wie in Kapitel 4.3.4 beschrieben, ist die Information über die Ladeleistung in 822 Ladevorgängen nicht enthalten und daher mussten diese für die Aufteilung anhand er Ladeleistung entfernt werden. Diese Datenreduktion symbolisiert der horizontale Strich über dem achten Modell in Abb. 31:.
Anschließend wurde im Modell 10 und 11 eine Kreuzvalidierung der Modelle durchgeführt, um das Modell hinsichtlich des vollständigen Datensatzes zu beurteilen. In Modell 12 wurde untersucht ob mittels einer Klassifikation unterschieden werden kann, ob es sich um einen Ladevorgang mit einer Ladeleistung größer bzw. kleiner als 5kW handelt. Abschließend wurde eine Parameteroptimierung mittels einer Rastersuche (engl. Grid Search) durchgeführt.

4.4.3.1 Modell 1: Vergleich der Ansätze

Die Ladevorgänge wurden, wie in Kapitel 4.3.3 beschrieben, mittels der drei Ansätze in ein maschinelles Lernproblem überführt. Anschließend wurden diese Ansätze in einem Modell der linearen Lasso Regression und des Zufallswaldes getestet. Es wurden die voreingestellten Parameter der linearen Lasso Regression verwendet, nur der Wert Alpha wurde auf 3 abgeändert [46].
In Abb. 32: ist das Ergebnis einer Modellbildung anhand von 101 Ladevorgängen ersichtlich. Hierzu wurde 79 Ladevorgänge zum Training und 22 zum Testen des Modelles verwendet. Die hellblaue durchgehende Linie stellt die Restladezeit, also die Soll-Größe dar. Die

Vorhersagen des Modelles sind hierbei in Grün, Grün strichliert und in Rot strichliert abgebildet. Die Vorhersage des ersten Ansatzes, jeden Zeitpunkt des Ladevorganges zu betrachten und somit die Sample Anzahl möglichst hoch zu halten, ist mit der dunkelgrünen durchgängigen Linie dargestellt und zeigt sehr sprunghaftes Verhalten. Das Modell für den ersten Ansatz wurde mit 70.527 Samples und 72 Features trainiert und getestet.
Der zweite Ansatz, ein Zeitfenster zu betrachten, ist durch die strichlierte grüne Linie abgebildet und zeigt eine bessere, stabilere Vorhersage als der erste Ansatz. Das Modell für den zweiten Ansatz wurde mit 2359 Samples und 142 Features trainiert und getestet.
Der dritte Ansatz, ein Zeitfenster mit beschreibender Statistik zu betrachten, ist mit der rot strichlierten Linie abgebildet und zeigt ein sehr ähnliches Verhalten als der zweite Ansatz. Das Modell für den dritten Ansatz wurde mit 2359 Samples und 217 Features trainiert und getestet.

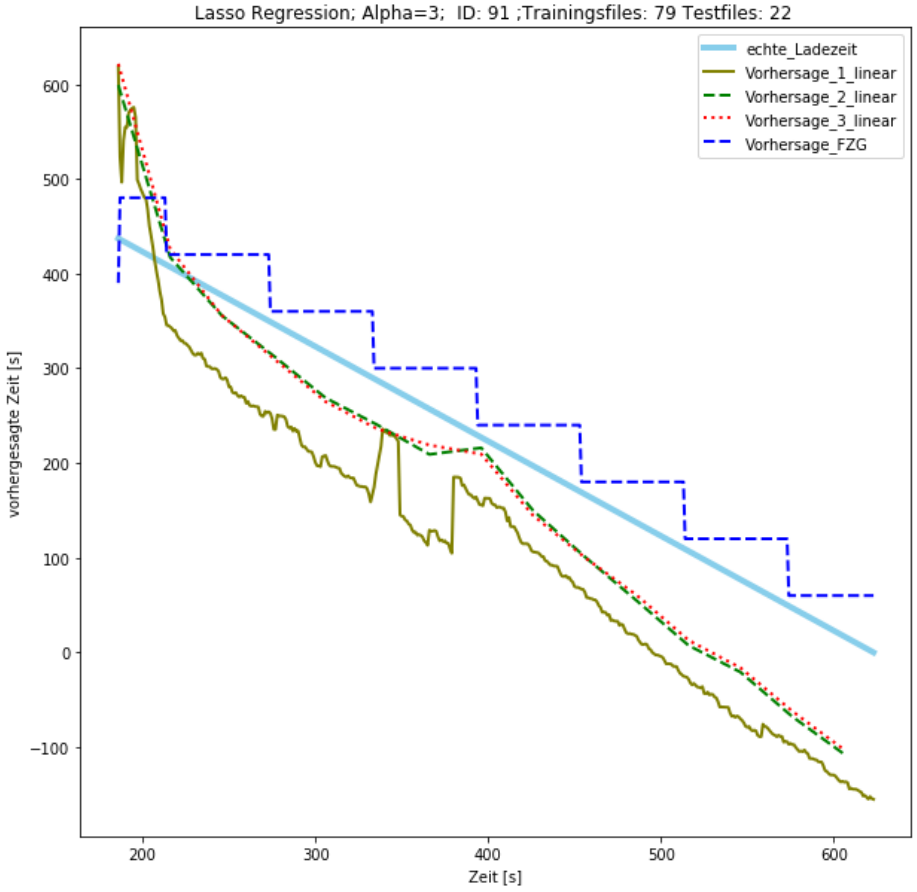

Abb. 32: Vergleich der Ansätze mittels Linearer Lasso Regression

Weiters ist die Prognose des bisherigen physikalischen Modelles in der dunkelblauen strichlierten Linie sichtbar. Die Ansätze 2 und 3, ein Zeitfenster zu betrachten, liefern eine wesentlich bessere und stabilere Vorhersage als der erste Ansatz, jeden Zeitpunkt zu betrachten. Auffällig ist, dass die Vorhersage unerwünschte negative Werte der Restladezeit beinhaltet, die physikalisch nicht möglich sind.

Die gleichen Trainings- und Testdaten wurden mit einem Random Forest Algorithmus trainiert und getestet. Hierbei wurden größtenteils die Standardparameter wie in [47] beschrieben verwendet. Die Anzahl der Bäume wurde auf 130 erhöht, da eine höhere Anzahl als 130

Bäume laut [48] keine signifikante Verbesserung mit sich bringt. Die Baumtiefe wurde versuchsweise auf 20 gesetzt um eine Generalisierung des Modelles zu gewährleisten. Die Vorhersagegenauigkeit der drei Ansätze ist in Abb. 33: abgebildet.

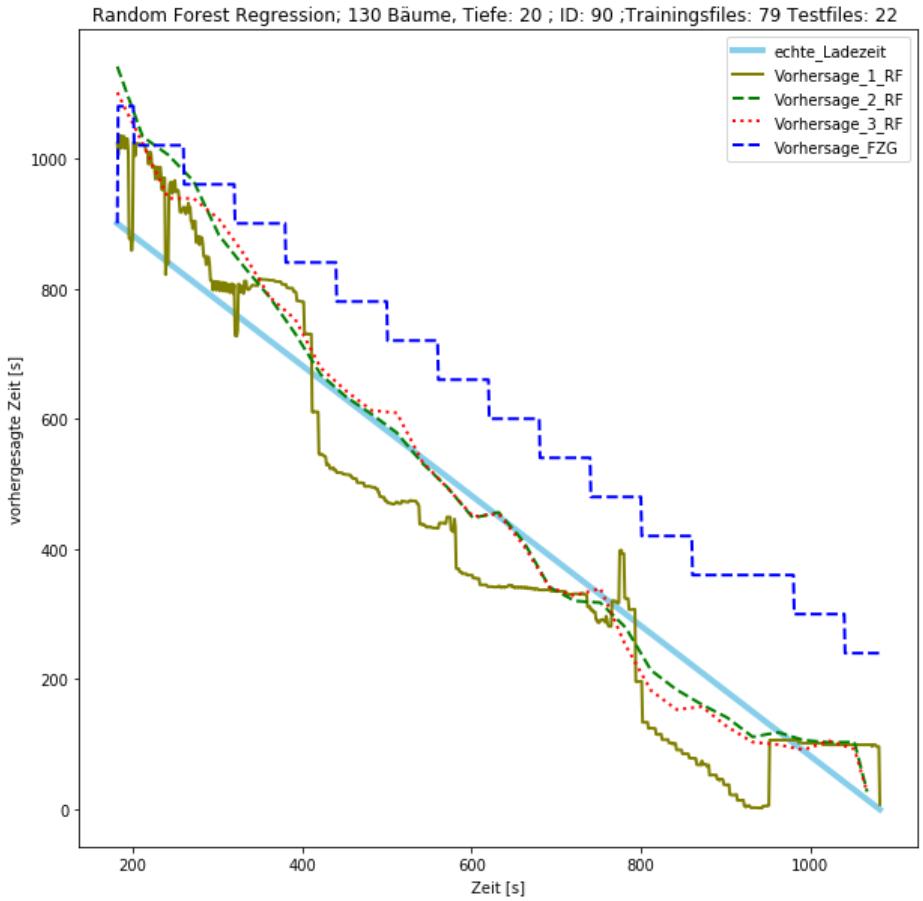

Abb. 33: Vergleich der Ansätze mittels Random Forest Regression

Deutlich sichtbar ist die bessere Vorhersage der Ansätze 2 (Zeitfenster / grün strichliert) und 3 (Zeitfenster mit beschreibender Statistik / rot strichliert) im Vergleich zum 1. Ansatz (Zeitpunkte / grün durchgehend).

In beiden Modellen LLR und RF hat der erste Ansatz, alle Zeitpunkte zu betrachten, schlechtere Ergebnisse geliefert und wird daher im Weiteren nicht mehr berücksichtigt. Die Genauigkeit der beiden Zeitfenster Ansätze 2 und 3 ist in der graphischen Auswertung sehr ähnlich bezüglich der Vorhersage. Die Berechnung des RMSE, wie in Gleichung 6 beschrieben, zeigte, dass der dritte Ansatz eine geringere Abweichung zur Soll Restladezeit aufweist. Der dritte Ansatz, ein Zeitfenster mit beschreibender Statistik, wurde aufgrund des geringeren RMSE Wertes, daher einer geringeren Abweichung und besserer Prognosegüte dem 2. Ansatz vorgezogen. Alle folgenden Modelle wurden mit dem Ansatz des Zeitfensters mit beschreibender Statistik trainiert und verglichen. Die Anzahl der Samples und Features der beiden gezeigten Modelle, RF und LLR, sind identisch.

4.4.3.2 Modell 2: Modellvergleich Lineare Lasso Regression - Random Forest

Im folgenden Abschnitt ist ein Vergleich der beiden Modelle Random Forest und der Linearen Lasso Regression dargestellt. Trainiert und getestet wurden die beiden Modelle, wie in 4.4.3.1 erklärt, mittels dem Ansatz, ein Zeitfenster mit beschreibender Statistik zu betrachten. In Abb. 34: ist die Gegenüberstellung der Prognosegenauigkeit der beiden Modelle dargestellt. Deutlich ersichtlich ist eine bessere Genauigkeit des RF Modelles, da die Prognose näher an der blauen Soll-Restladezeit liegt. Weiters ist ersichtlich, dass das LLR Modell negative Restladezeiten prognostiziert. Dies ist physikalisch nicht möglich. Diese negativen Restladezeiten werden vom RF Modell nicht prognostiziert.

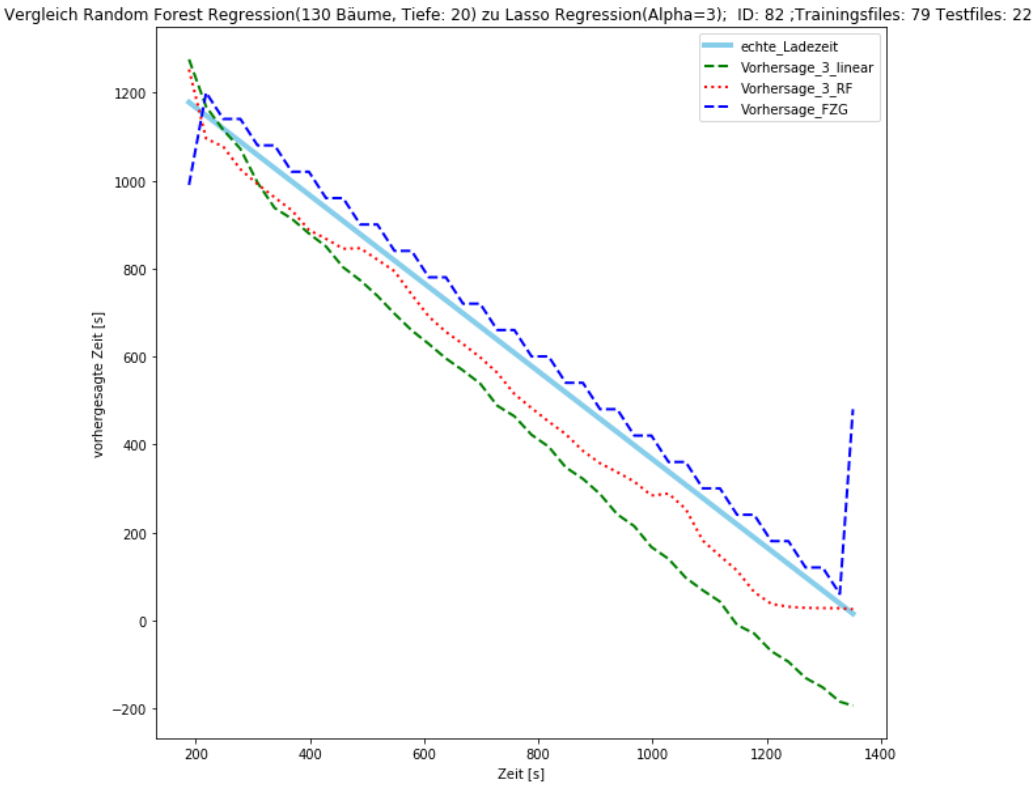

Abb. 34: Modellvergleich Lineare Lasso Regression - Random Forest

Die Abweichung des RMSE Wertes des RF Modelles war wesentlich geringer. Die drei erwähnten Vorteile waren in allen Testdaten sichtbar und aus diesem Grund wurde das RF Modell für alle folgenden Modellbildungen ausgewählt. Zusätzlich ist in Abb. 34: noch die Vorhersage des bereits im Fahrzeug implementierten physikalischen Modelles in dunkelblau sichtbar. Hierzu wurden 2359 Samples und 217 Features zur Modellbildung und Testung herangezogen.

4.4.3.3 Modell 3: Eignung maschinelles Lernen für Batterieladezeit

Im Folgenden wurde die grundlegende Frage untersucht, ob es möglich ist, mittels maschinellem Lernen ein Modell mit guter Präzision zur Vorhersage der Batterieladezeit zu

bilden. In Abb. 16: wurde aufgezeigt, dass das bisherige physikalische Modell, in 17% der Ladevorgänge eine gute Prognose der Restladezeit liefert. Diese Daten wurden ausgewählt, unter der Annahme, dass sie ein ähnliches physikalisches Verhalten aufweisen und daher eine gute Grundlage für die Beantwortung der oben genannten Frage bilden. Das RF Modell wurde mit diesen 536 Ladevorgängen trainiert und getestet. Hierbei wurde der Ansatz der Zeitfenster mit beschreibender Statistik verwendet, wie in Kapitel 4.3.3.3 erklärt. Zur Bildung des Modelles wurden 51.279 Samples und 217 Features herangezogen. Das Bewertungskriterium dabei war die Anfangsgenauigkeit der Prognose, wie in Kapitel 4.4.2.1 bereits erläutert. Ist die Abweichung der Prognose kleiner als ±7% zum Sollwert, wird das Bewertungskriterium erfüllt. In Abb. 35: wird dies grafisch dargestellt. Hierbei ist das 7% Kriterium in orange dargestellt, bezogen auf die Anfangsrestladezeit. In blau ist die Soll-Restladezeit und in Grün die Vorhersage des RF Modelles abgebildet.
Das Modell wurde mit 80% der Daten trainiert und mit den restlichen 20% getestet. Die Anzahl der Bäume im RF wurde auf 130 gesetzt und die Baumtiefe auf 20 begrenzt. Alle restlichen Parameter wurden nicht verändert und sind wie in [47] beschrieben.

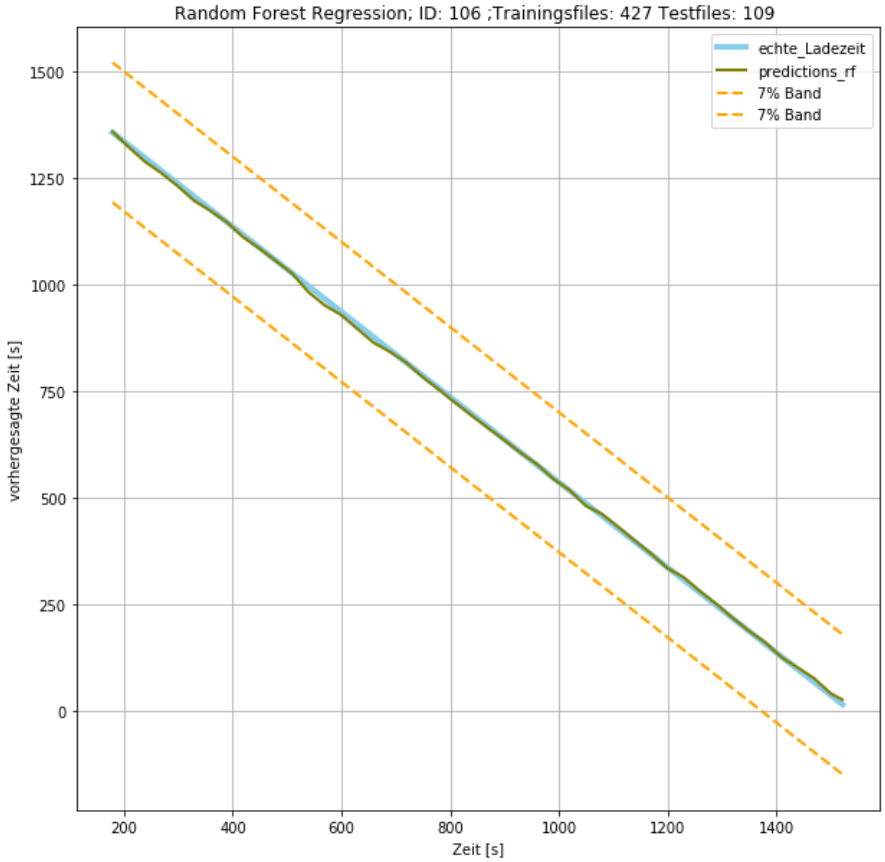

Abb. 35: Modell 3: RF Modell – Ist eine Vorhersage möglich mittels maschinellem Lernen?

In 108 der 109 Testdaten erfüllte die Prognose des RF Modelles das Bewertungskriterium, welches die grundlegende Frage mit Ja beantwortet, ob mittels maschinellem Lernen eine Prognose der Batterieladezeit möglich ist.

Eine Testdatei, sprich ein Ladevorgang, erfüllte das Bewertungskriterium nicht. Diese Vorhersage des RF Modelles lag außerhalb der 7% Grenze. Im Folgenden wurde untersucht, warum dieser Ladevorgang das Bewertungskriterium nicht erfüllt.
In Abb. 36:, im Diagramm links oben, ist die Vorhersage dieses Ladevorganges im Vergleich zur Soll-Restladezeit abgebildet. Hierbei ist ersichtlich, dass die Vorhersage des Modelles komplett daneben liegt. Es wird vom Modell eine Restladezeit von nahezu 0 Sekunden vorhergesagt.
Die restlichen 3 Diagramme in Abb. 36: zeigen die 3 wichtigsten Features, die das RF Modell zur Vorhersage erlernt hat. Diese sind der Batteriestrom, die Batteriespannung und der Batterieladezustand. In grün ist hierbei jeweils ein Ladevorgang abgebildet, dessen Prognose des RF Modelles das Bewertungskriterium erfüllt. In blau ist dieser eine Ladevorgang abgebildet, dessen Prognose des RF Modelles das Bewertungskriterium nicht erfüllt.
Der Batterieladezustand sowie die Batteriespannung bleiben konstant auf 90% beziehungsweise auf ca. 405 Volt. Deutlich sichtbar ist der Batteriestrom der im Bereich von 0 Ampere schwankt. Die Batterie wird folglich mit enorm kleiner Leistung über den ganzen Zeitbereich geladen. Ein solches Verhalten ist in keinem anderen Ladevorgang des Test- bzw. Trainingsdatensatzes abgebildet und kann deswegen vom RF Modell nicht vorhergesagt werden.

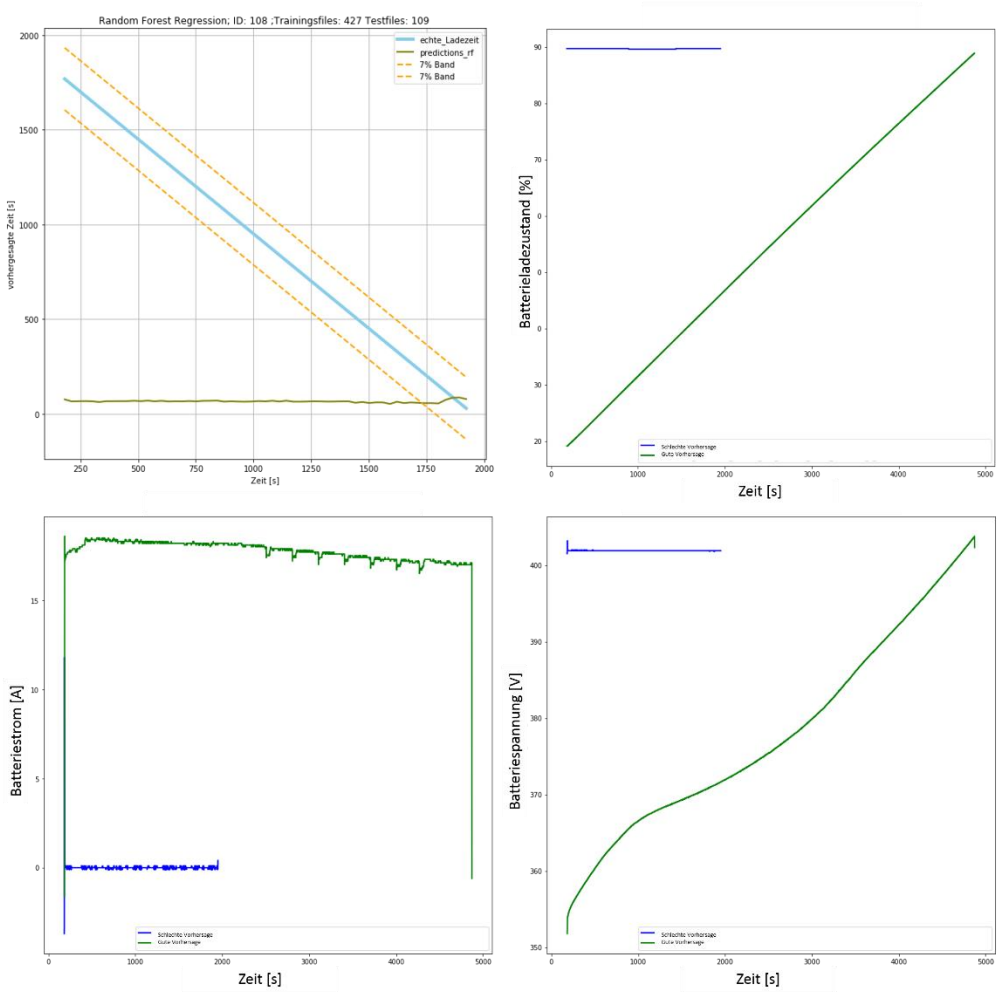

Abb. 36: Modell 3: Datenanalyse der Modellvorhersage

4.4.3.4 Modell 4: Anwenden des 1. Modelles auf den restlichen Datensatz

Wie in Kapitel 4.4.3.3 erläutert, wurden zum Bilden des 1. Modelles nur 17% der Gesamtdaten verwendet. Durch die guten Ergebnisse dieses Modelles, kam die Idee, dieses Modell auf die restlichen 83% der Daten anzuwenden, um daraus etwaige Abnormitäten und Abweichungen in den Daten zu erkennen.
Somit wurde ein RF mit 536 Ladevorgängen trainiert und mit den restlichen 2437 Ladevorgängen getestet. Dies ist ein unübliches Verhalten im Bereich des maschinellen Lernens, da in allen Lehrbüchern von maschinellem Lernen eine Trainings- und Testdaten Aufteilung von 80% zu 20% empfohlen wird, siehe [35] und [37]. In Abb. 37: links ist eine grafische Evaluierung dieses Modelles abgebildet. Hierbei ist auf der x-Achse der Ist-Wert der Batterieladezeit der Testsamples aufgetragen und auf der y-Achse die Vorhersage der Batterieladezeit des Modelles. Im Optimalfall liegen alle blau dargestellten Wertepaare auf der schwarz strichlierten 45° Linie. Deutlich sichtbar sind starke Abweichungen der Vorhersage.

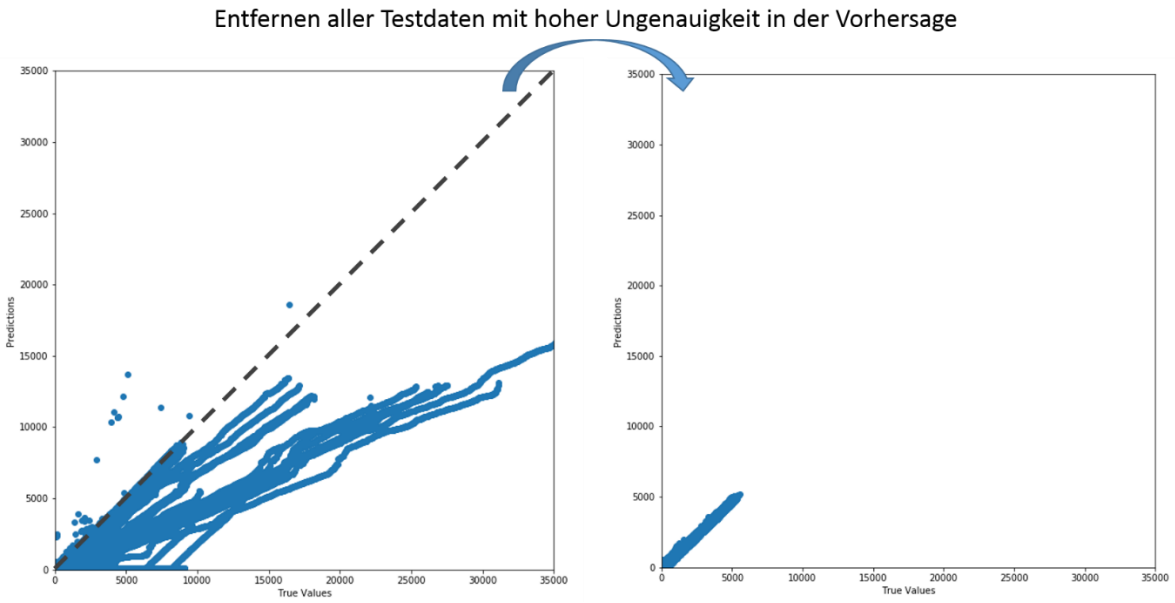

Abb. 37: Modell 4: Vergleich der vorhergesagten Werte zu den Ist-Werten

Im rechten Bild wurde das gleiche Modell erneut getestet, allerdings mit einem geringeren Testdatensatz. Es wurden alle Testdaten entfernt, deren Vorhersage eine größere Abweichung als 7% aufwies und diese entfernten Daten wurden anschließend auf Gemeinsamkeiten untersucht. In Abb. 37: ist ersichtlich, dass Samples mit hoher Batterieladezeit schlecht prognostiziert werden und je größer diese Zeit wird, desto größer wird die Abweichung der Prognose des Modelles. Es wurde die Genauigkeit des Modelles aufgrund der grafischen Übereinstimmung und des Bestimmtheitsmaßes R^2 bewertet.
In diesem Modell wurde ein R^2 Wert von 0,99 auf den Testdatensatz erreicht und ein R^2 Wert von 0,31 auf den Testdatensatz.
Im Folgenden werden diese entfernten Daten untersucht. Hierbei wird evaluiert, wieviel Zeit benötigt wird, um den Ladezustand der Batterie um 10% zu erhöhen. Wie in Kapitel 4.2.1 erläutert, startet jeder Ladevorgang bei einem unterschiedlichen Ladezustand. Um alle Ladevorgänge vergleichen zu können, muss ein Bereich des Ladezustandes gefunden werden, der in allen Daten repräsentiert wird. Der Bereich des Ladezustandes zwischen 75 und 85% ist am häufigsten in den Daten abgebildet und wurde daher für den Vergleich gewählt. In Abb. 38: wird gezeigt, wie viel Zeit benötigt wird, um die Batterie von 75% auf 85% Ladezustand zu laden. In grün sind hierbei die Ladevorgänge abgebildet, dessen Prognose des bisherigen physikalische Modelles innerhalb des Bewertungskriteriums liegt. In orange sind die Ladevorgänge abgebildet, dessen Prognose des bisherigen physikalischen Modelles außerhalb des Bewertungskriteriums liegt. Diese Aufteilung wurde gewählt, um etwaige Abhängigkeiten und Zusammenhänge in den Daten zu erkennen.
Wie in Kapitel 4.2 beschrieben, handelt es sich in den Daten um Hybridfahrzeuge mit einer Batteriekapazität von 11,5 kWh. Die maximale Leistung des verbauten Ladegerätes beträgt 7,2kW. Die maximal erreichbare Ladeleistung, inklusive der Verluste der Umwandlung von Wechselstrom auf Gleichstrom und Versorgung der Nebenverbraucher liegt bei 6,7-6,8 kW.

Sind zusätzliche Verbraucher, wie Heizung oder Klimaanlage aktiv, reduziert sich die Energie, die in der Batterie ankommt dementsprechend.
In der Datenanalyse in Kapitel 4.3.1 wurde ersichtlich, dass die Batterie nur bis 90% geladen wird und somit die verwendbare Kapazität der Batterie auf 10,3 kWh sinkt. Berücksichtigt man dies, werden mindestens 90 Minuten zur vollständigen Ladung der Batterie benötigt. Daraus lässt schließen, das eine Anhebung des Ladezustandes um 10% im Optimalfall ca. 11 Minuten dauert. In Abb. 38: ist ersichtlich, dass ein Großteil der Ladevorgänge zwischen 630 und 800 Sekunden benötigt um die Batterie 10% zu laden. Dies sind umgerechnet zwischen 10,5 und 13 Minuten. Weiters ist ersichtlich, dass ein kleiner Teil der Daten, weitaus mehr Zeit benötigt um die Batterie um 10% Ladezustand zu laden. Dies ist durch eine flachere Steigung in Abb. 38: ersichtlich.

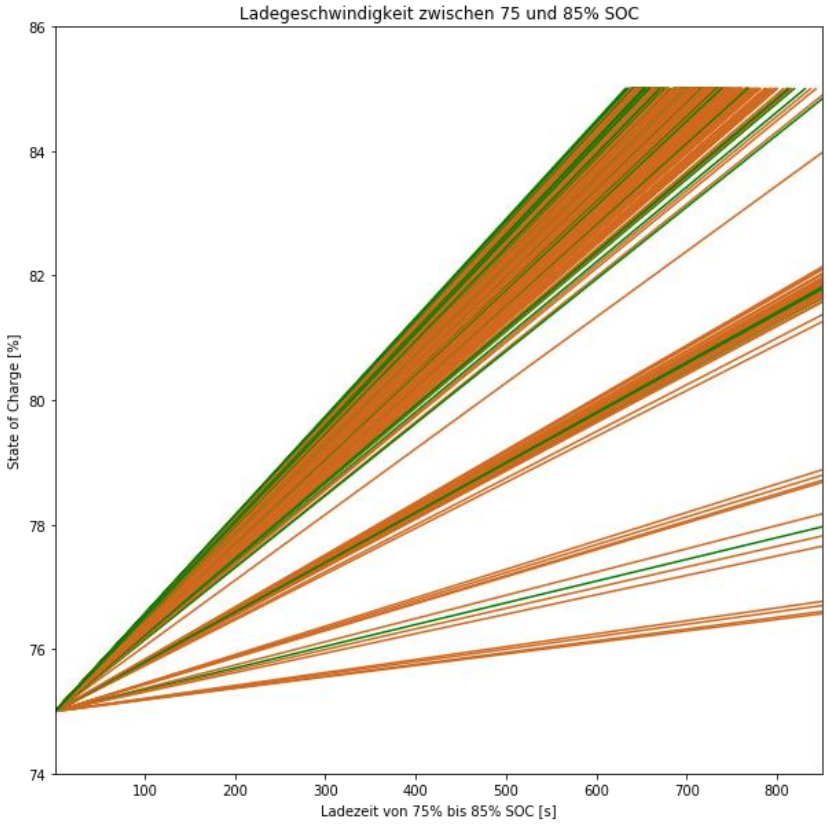

Abb. 38: Datenanalyse: Ladegeschwindigkeit

Aus diesem Grund besteht die Annahme, die Daten anhand der Ladedauer zu unterteilen. Dies wird im folgenden Abschnitt 4.4.3.5 überprüft.

4.4.3.5 Modell 5: RF Modell angewendet auf alle Daten

Im Abschnitt 4.4.3.4, wurde das Modell unkonventioneller Weise mit weitaus mehr Testdaten getestet als es mit Trainingsdaten trainiert wurde, um daraus neue Schlüsse zu ziehen. Um diese Schlussfolgerungen zu überprüfen, wurde in diesem Abschnitt ein Modell gebildet, mittels üblicher 80% / 20% Aufteilung zwischen Trainings- und Testdaten. Hierbei wurden alle

verfügbaren 3131 Datensätze verwendet. Dies ergibt 358.948 Samples und 169 Features. Um eine einheitliche Basis an Signalen zu schaffen, wurden alle Signale entfernt, die nicht in allen Ladevorgängen präsent sind. Dies musste gemacht werden, da wie in Kapitel 4.2.2 erwähnt, je nach Version des Datenloggers die aufgezeichneten Signale in den Ladevorgängen variieren. Dabei fallen wichtige Signale wie Batteriespannung, Klimaanlagenstrom, verwendete Phasen des Ladegerätes, Leistung des Ladegerätes, Batteriestrom sowie Heizungsstrom weg. In diesem Abschnitt wurde versucht, möglichst alle Ladevorgänge miteinzubeziehen und daher die Anzahl an Samples möglichst groß zu halten. Hierbei wurde ein R^2 Wert von 0.99 auf den Trainingsdaten erreicht und ein R^2 Wert von 0.87 auf den Testdatensatz. Betrachtet man nun die grafische Interpretation der Ergebnisse in Abb. 39:, ist ersichtlich, dass es teilweise sehr große Abweichungen gibt. Hierbei sind auf der y-Achse die Vorhersagen des RF Modelles aufgetragen und auf der x-Achse die Soll-Werte der Restladezeit. Im Optimalfall sind diese Wertepaare gleich groß und liegen daher auf der schwarz strichlierten 45° Linie.

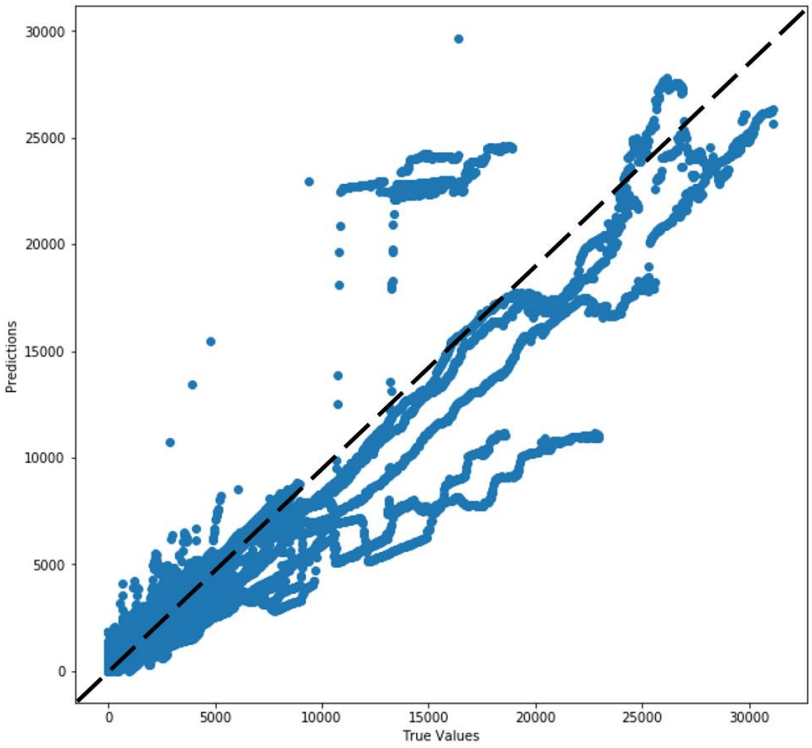

Abb. 39: Modell 5: Vorhersageüberlick über alle Ladevorgänge

Um eine Aussage treffen zu können, wurden anschließend einzeln die Ladevorgänge untersucht und auf Gemeinsamkeiten überprüft. Betrachtet man nun beispielhaft die Vorhersage von 4 Ladevorgängen näher im Detail, wie in Abb. 40: abgebildet, wird ersichtlich, dass die Batterieladezeit in den Ladevorgängen 1 und 2 sehr gut prognostiziert wird und sehr nahe an der Soll Restladezeit liegt. In den Ladevorgängen 3 und 4 ist die Vorhersage des RF Modelles sehr sprunghaft und die Abweichung der Prognose zur Soll – Restladezeit ist teilweise sehr hoch. Der Batterieladevorgang startet in allen 4 Diagrammen zum Zeitpunkt 0

der x-Achse. Die Restladezeit der Batterie ist in hellblau ersichtlich. Erreicht die hellblaue Linie den Zeitpunkt 0 der y-Achse, ist der Ladevorgang beendet und die Batterie vollgeladen.
Nach Durchsicht der Rohdaten des Ladevorganges 3 in Abb. 40:, lag die Leistung des Ladegerätes bei ca. 0,7 kW und die Batterie wurde von 89% Ladezustand auf 90% realen Batterieladezustand geladen.
Der Ladevorgang 4 wurde mit einer Leistung von 3,6 kW geladen von 52% auf 88,5% realen Batterieladezustand.
Im Ladevorgang 1 wurde mit einer Leistung von 7,2 kW von 25,2% auf 88,9% Batterieladezustand geladen. Im Ladevorgang 2 wurde ebenfalls mit einer Leistung von 7,2 kW geladen und der reale Batterieladezustand wurde von 18,7% auf 89,4% geändert.

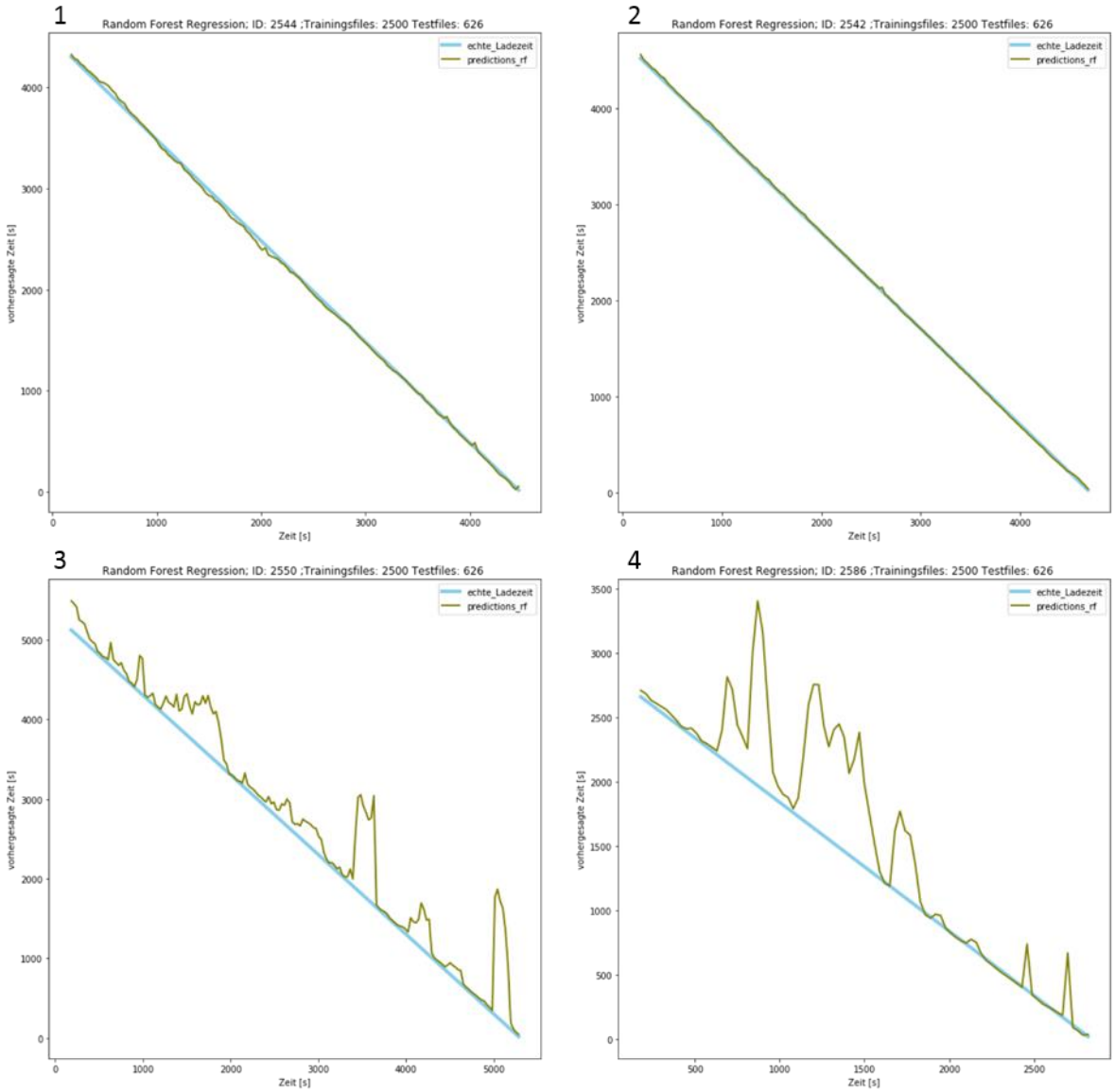

Abb. 40: Modell 5: Prognosegüte anhand 4 Beispielladevorgängen

Auffällig dabei ist, dass Ladeleistungen über 5 kW sehr gut vom Modell prognostiziert werden und bei Ladeleistungen kleiner als 5kW, kommt es zu hohen Schwankungen. Somit besteht

die Annahme die Daten anhand der Ladeleistung zu unterteilen. Will man die Ladevorgänge anhand der Ladeleistung unterteilen, muss dieses Signal in allen Daten vorhanden sein. Dies ist wie in Kapitel 4.3.4 beschrieben nur in 2309 Ladevorgängen der Fall. Die Unterteilung anhand der Ladedauer ist jedoch mit allen Daten möglich. Um die Anzahl der Ladevorgänge und somit die Anzahl der Samples möglichst hoch zu belassen, wurden die Ladevorgänge zuerst in den folgenden Abschnitten anhand der Ladedauer unterteilt.

4.4.3.6 Modell 6: RF Modell Unterteilung bezüglich der Ladedauer < 6000s

Hierbei werden zur Modellbildung nur Ladevorgänge herangezogen, die eine kürzere Ladedauer als 6000 Sekunden, also 100 Minuten aufweisen. Indirekt ist dies auch eine Einteilung bezüglich der Ladeleistung, da eine kurze Ladedauer auch eine hohe Ladeleistung impliziert. Dies stimmt jedoch nicht immer, da manche Ladevorgänge bei beispielsweise 70% Ladezustand starten und somit führt auch eine geringe Leistung zu einer Ladezeit unter 100 Minuten, bei einer Vollladung der Batterie.
Die Zeitreihen wurden, wie in Kapitel 4.3.3.3 beschrieben, mit dem Ansatz der Zeitfenster mit beschreibender Statistik in ein Maschinelles Lernproblem überführt.
Um eine einheitliche Basis an Signalen zu schaffen, wurden alle Signale entfernt, die nicht in allen Ladevorgängen präsent sind. Dies musste gemacht werden, da wie in Kapitel 4.2.2 erwähnt, je nach Version des Datenloggers die aufgezeichneten Signale in den Ladevorgängen variieren. Dabei fallen wichtige Signale wie Batteriespannung, Klimaanlagenstrom, verwendete Phasen des Ladegerätes, Leistung des Ladegerätes, Batteriestrom sowie Heizungsstrom weg. Es war die Intention, die Anzahl der Samples möglichst hoch zu halten unter der Annahme, das wegfallenden Signale eventuell durch andere, noch vorhanden Signale abgebildet sind und daher für eine erfolgreiche Modellbildung nicht essentiell sind.
Der RF wurde mit Standardparametern gebildet, bis auf Abänderungen bezüglich der Baumanzahl und der Baumtiefe. Diese wurden wie bisher auf 130 und auf 20 gesetzt. Das Modell wurde mit 296.608 Samples und 169 Features trainiert und getestet. Diese Samples stammen aus 3023 Ladevorgängen. In Abb. 41: wird links ein Soll-Ist Vergleich der Vorhersage des Modelles über alle Testdaten dargestellt und rechts werden die wichtigsten 4 Features des RF Modelles gezeigt. Sind die vorhergesagten Werte und die echten Werte gleich groß, liegen sie auf der schwarz abgebildeten 45 Grad Gerade.

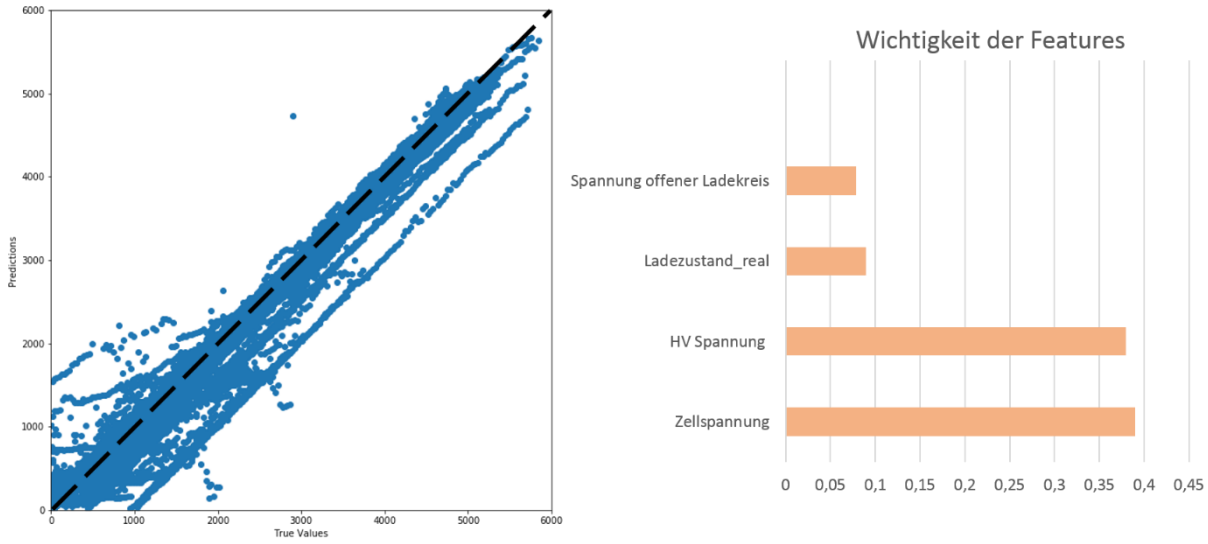

Abb. 41: Modell 6: Übersicht

Wenn man Abb. 39: und Abb. 41: vergleicht, ist ein deutlicher Unterschied erkennbar. Die Wertepaare der Vorhersage und der Testwerte liegen näher an der schwarz strichlierten Soll Gerade. Der R^2 Wert liegt in diesem Modell bei den Testdaten bei 0,99 und bei den Trainingsdaten bei 0,90.
In der rechten Darstellung sind die wichtigsten 4 Features des RF Modelles abgebildet. Das wichtigste Feature ist die Zellspannung mit einer Gewichtung von 0.39. Die Batteriespannung des Hochvolt Kreises ist das zweitwichtigste Feature mit einer Gewichtung von 0,38. Der reale Ladezustand der Batterie folgt an dritter Stelle mit einer Gewichtung von 0,1. Das letzte Feature, in Abb. 41: als „Spannung offener Ladekreis“ bezeichnet, ist eine Spannung die vom Steuergerät berechnet wird. Diese Spannung würde anliegen, wenn das Ladegerät abgeschlossen und die Ladung beendet wird, bei diesem aktuellen Ladezustand. Um die Batterie zu laden muss eine höhere Spannung angeschlossen werden, als die Batteriespannung. Wird das Ladegerät abgeschlossen, sinkt die Batteriespannung. Diese, vom Steuergerät voraus berechnete Spannung, falls das Ladegerät in dem Moment abgeschlossen werden würde, wird als „Spannung offener Ladekreis“ bezeichnet.

4.4.3.7 Modell 7: RF Modell Unterteilung bezüglich der Ladedauer > 6000s

Hierbei wurden Ladevorgänge herangezogen, die eine längere Ladedauer als 100 Minuten aufweisen. Die Zeitreihen wurden, wie in Kapitel 4.3.3.3 beschrieben, mit dem Ansatz der Zeitfenster mit beschreibender Statistik in ein maschinelles Lernproblem überführt. Es wurden die gleichen Signale wie in Abschnitt 4.4.3.6 entfernt, da diese nicht in allen Datensätzen präsent waren.
Es wurden die gleiche Parametereinstellungen verwendet als in Kapitel 4.4.3.6. Das Modell wurde mit 36.328 Samples und 169 Features trainiert und getestet. Diese Samples wurden aus 108 Ladevorgängen gebildet. Der R^2 Wert bezogen auf die Trainingsdaten ist 0,99 und bezogen auf die Testdaten bei 0,62. Die Vorhersagegenauigkeit der Testdaten und die Wichtigkeit der Features ist in Abb. 42: abgebildet.

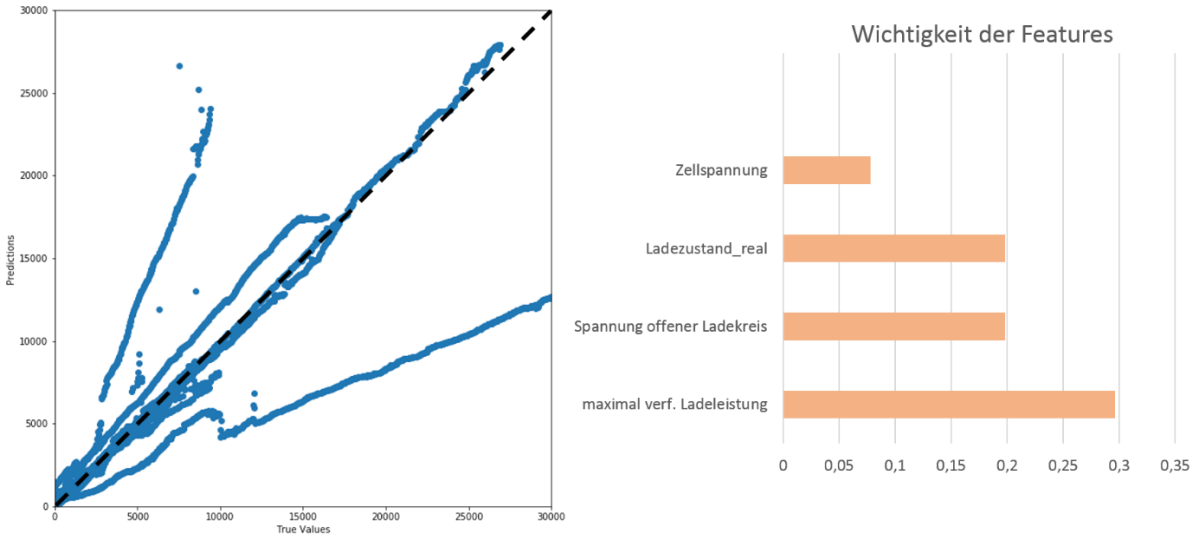

Abb. 42: Modell 7: Übersicht

Ersichtlich ist, dass es zu weitaus größeren Abweichungen kommt, hinsichtlich der vorhergesagten Werten, als in Kapitel 4.4.3.6. Es werden zu große sowie auch zu kleine Restladezeiten der Batterie vorausgesagt. Punkte die nahezu eine Linie bilden, sind zu einem Ladevorgang gehörig. Ersichtlich hierbei ist, dass der ganze Zeitbereich eines Ladevorganges falsch vorhergesagt wird, nicht nur ein Zeitpunkt oder Sample.
Die maximal verfügbare Ladeleistung ist hierbei mit einer Gewichtung von 0,29 der wichtigste Parameter. Dies ist die maximal abgebbare Leistung der Ladeinfrastruktur und nicht die tatsächlich aufgenommene Leistung des Ladegerätes. Die aktuelle Ladeleistung musste als Feature entfernt werden, da diese nicht in allen Datensätzen präsent war. Nach Untersuchung der Rohdaten, wurde ersichtlich, dass in manchen Fällen die tatsächliche Leistung zur maximalen Leistung der Infrastruktur abweicht. Daher wird vom Modell eine Information zur Bildung des Modelles herangezogen, das nicht in allen Fällen der bezogenen Leistung entspricht.
Es ist es durchaus schlüssig, dass die Ladeleistung in diesem Modell als wichtigstes Feature herangezogen wird, da es eine höhere Variation der Leistungen in den Daten gibt. Der reale Ladezustand sowie die Spannung des offenen Ladekreises sind die zweit- und drittwichtigsten Parameter mit einer Gewichtung von 0,19. Die Zellspannung jeder Batteriezelle wird mit 0,07 als letztes der 4 wichtigsten Features angeführt.

4.4.3.8 Modell 8: RF für Ladevorgänge > 5kW

Um die Ladevorgänge anhand der Ladeleistung, wie in Kapitel 4.3.5 beschrieben, zu unterteilen muss die Info über die Ladeleistung in allen Daten vorhanden sein. Bisher wurde in Abschnitt 4.4.3.6 und 4.4.3.7 versucht, die Anzahl der Samples möglichst hoch zu halten. Hierbei wurden alle Ladevorgänge als Datengrundlage herangezogen und die Features auf die maximal mögliche, in allen Daten präsente, Anzahl reduziert. Um Kernsignale wie Klimaanlagenstrom, verwendete Phasen des Ladegerätes, Leistung des Ladegerätes, oder

Heizungsstrom in den Features inkludiert zu haben, muss der Datensatz verkleinert werden, wie in Kapitel 4.3.4 beschrieben. Somit ist eine Unterteilung anhand der Ladeleistung möglich. Die Anzahl der Ladevorgänge über 5kW beträgt somit 2067. Diese bilden die Basis für eine Anzahl von 229.465 Samples mit 190 Features. Mit diesen Samples wurde nach einer Aufteilung von 80% Trainingsdaten und 20% Testdaten das RF Modell trainiert und getestet. Der R^2 Wert bezogen auf die Trainingsdaten liegt bei 0,99 und bezogen auf die Testdaten bei 0,98.

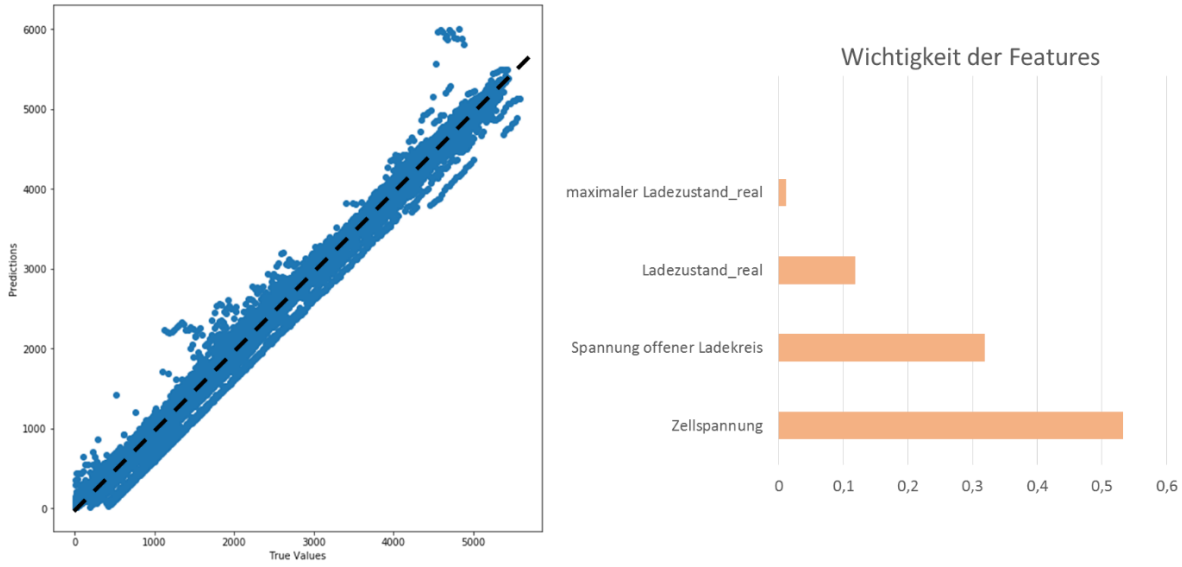

Abb. 43: Modell 8: Überblick

In Abb. 43: ist links die Vorhersage der echten Restladezeit der Batterie gegenübergestellt. Deutlich sichtbar ist die verbesserte Annäherung an die schwarz strichlierte Linie, im Vergleich zu den bisherigen Modellen. In Abb. 43: rechts ist ersichtlich, dass das wichtigste Feature die Zellspannung mit einer Gewichtung von 0,53 ist. Als nächstes Feature wird die Spannung des offenen Ladekreises vom Modell als wichtig erachtet mit einer Gewichtung von 0,32. Abschließend wird der reale Ladezustand mit einer Gewichtung von 0,12 und der maximale reale Ladezustand mit 0,03 gewichtet.

4.4.3.9 Modell 9: RF für Ladevorgänge < 5kW

Wie in Abschnitt 4.4.3.8 beschrieben, wurde hier die Datengrundlage verringert um alle wichtigen Features inkludieren zu können. Die Anzahl der Ladevorgänge unter 5kW beträgt somit 242. Diese bilden die Basis für eine Anzahl von 52.902 Samples und 190 Features. Mit diesen Samples wurde nach einer Aufteilung von 80% Trainings- und 20% Testdaten das RF Modell trainiert und getestet. Das Ergebnis dieser Vorhersage wird in Abb. 44: dargestellt.

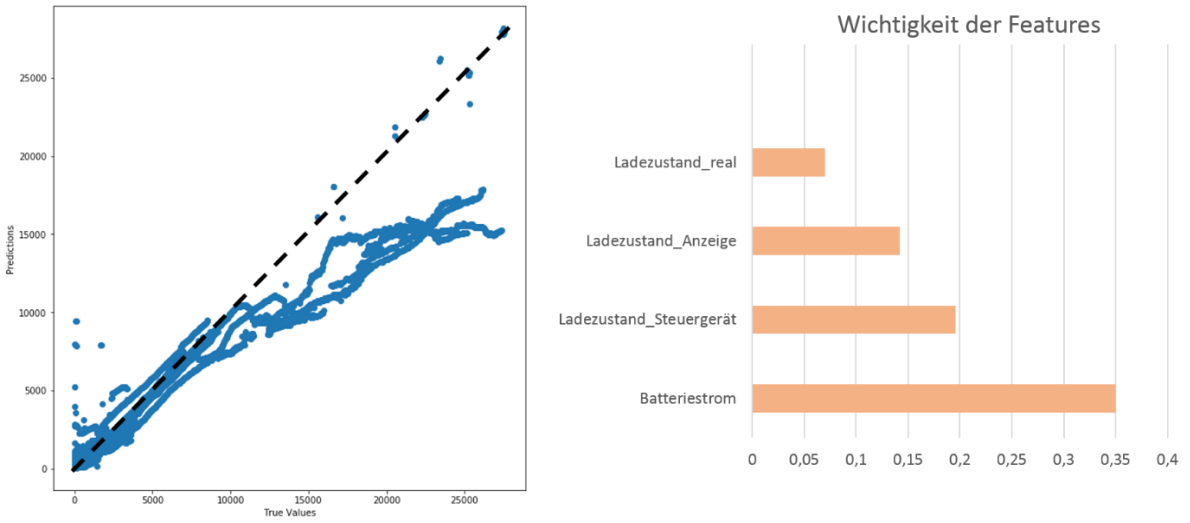

Abb. 44: Modell 9: Übersicht

Deutlich ersichtlich ist die Abweichung zur schwarzen Soll-Gerade. Weiters ist interessant, dass im Vergleich zum Modell 7 die Ladeleistung nicht in den wichtigsten Features vorhanden ist. Der Batteriestrom gibt auch Auskunft, mit welcher Leistung geladen wird. Die restlichen 3 Features geben Auskunft über den Ladezustand, wie in Kapitel 4.3.1 genau erläutert.
Der R^2 Wert bezogen auf die Trainingsdaten liegt bei 0,99 und bezogen auf die Testdaten bei 0,83. Die Kombination der Modelle 5 und 6 bildet bisher das beste Ergebnis, bei voriger Unterteilung hinsichtlich der Ladeleistung.

4.4.3.10 Modell 10: Kreuzvalidierung Modell 8

Um zu überprüfen ob das Modell auf alle Daten eine gute Prognosegüte liefert, wurde eine Kreuzvalidierung angewandt, wie in Kapitel 3.3.4.3 beschrieben.
Die Anzahl der Samples liegt bei. 229.465 Ladevorgängen die je nach Runde in ca. 182.000 bis 185.000 Trainingsbeispielen und 44.000 bis 47.000 Testbeispiele aufgeteilt werden. Anschließend werden die Modelle mittels RMSE verglichen und ausgewählt. In Abb. 45: wird das grafische Ergebnis der Kreuzvalidierung aufgezeigt. Hierbei ist auffällig, dass es in Runde 1 zu weitaus größeren Abweichungen der Prognose kam, als in den restlichen Runden. Die Prognosegüte in den restlichen Runden 2-5 ist ähnlich hoch.
Bewertet man nun alle Vorhersagen der Runden 1-5, wie in Kapitel 4.4.2.1 erläutert, erfüllt die Prognose in 80,1% der Fälle das Bewertungskriterium. Dies entspricht 1674 Ladevorgängen. Der RMSE Wert, bezogen auf alle Runden 1-5 liegt bei 120,3. Wie In Abb. 45: ersichtlich, kommt es in Runde 1 zu größeren Abweichungen bezüglich der Restladezeitvorhersage. Um zu verstehen, warum es bei diesen Ladevorgängen zu starken Abweichungen kommt, werden diese im Folgenden genauer untersucht.

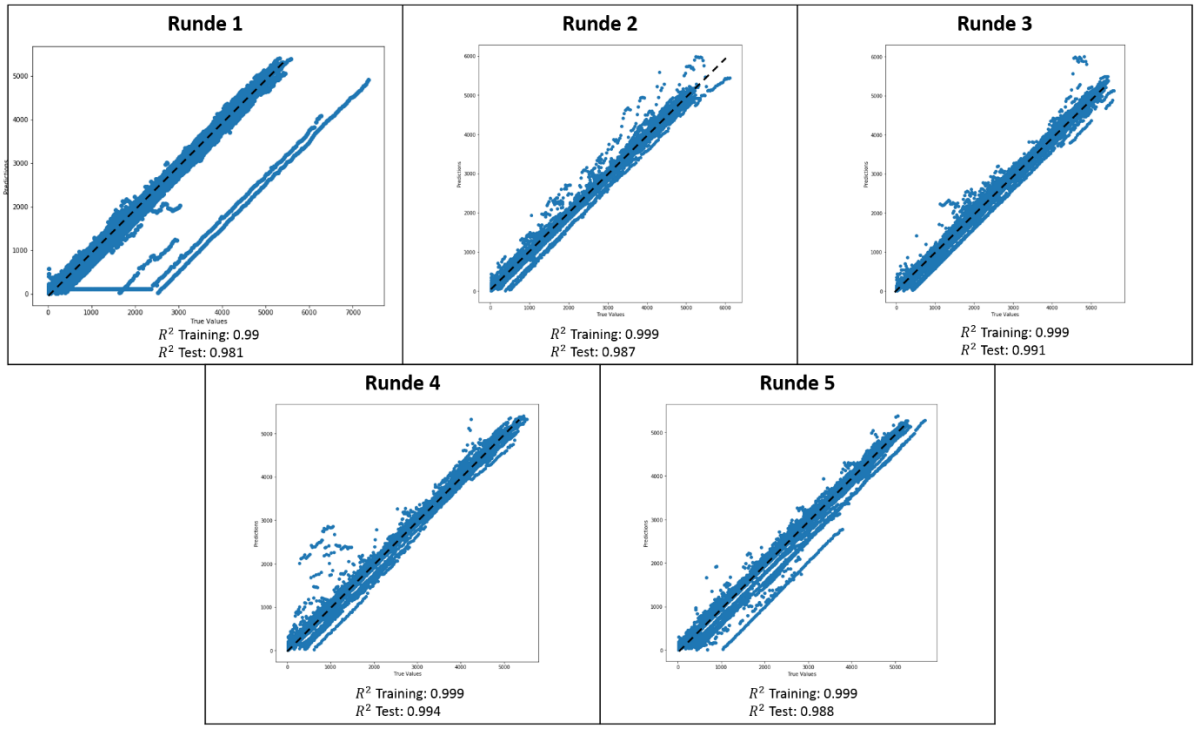

Abb. 45: Ergebnisse Kreuzvalidierung Ladeleistung größer 5kW

Im Folgenden werden die Ergebnisse der Runde 1 untersucht, um festzustellen ob die Prognose des Modelles schlüssig ist. In Abb. 46: sind in Rot, Grün und Beige die drei Ladevorgänge markiert, die zu starken Abweichungen geführt haben. Diese 3 Vorgänge werden hinsichtlich der wichtigsten 2 Features betrachtet und mit einem Ladevorgang, der gut vom Modell, prognostiziert wurde, verglichen.

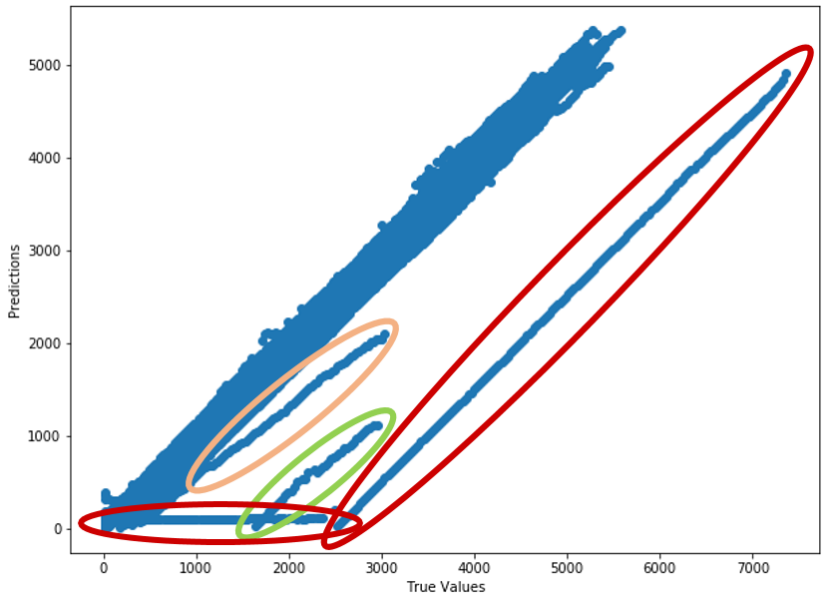

Abb. 46: Untersuchung der Ergebnisse - Kreuzvalidierung

Die zwei wichtigsten Features dieses Modelles sind die Zellspannung, sowie der Ladezustand. In Abb. 47: werden somit diese zwei Features grafisch dargestellt anhand der drei, in Abb. 46: dargestellten Ausreißer, und mit einem Ladevorgang verglichen, der vom Modell gut prognostiziert wird. Dieser Ladevorgang ist in Schwarz in Abb. 47: dargestellt. Die restlichen 3 Ladevorgänge, in grün, rot und beige dargestellt bilden farblich die gleichen Ladevorgänge wie in Abb. 46: ab.
Betrachten wir nun den ersten Ladevorgang, in Rot dargestellt. Hierbei wird zuerst normal geladen, anschließend wird etwas Energie aus der Batterie genommen und dann wird der Ladezustand bei annähernd 90 % gehalten. Zu diesem Zeitpunkt wurde die Vorklimatisierung des Fahrzeuges eingeschaltet und da die Leistung der Vorklimatisierung größer ist, als das Ladegerät liefern kann, sinkt der Batterieladezustand. Sobald die eingestellte Innentemperatur des Fahrzeuges erreicht wird, wird die Temperatur nur mehr gehalten und es wird weniger Leistung benötigt, somit kann auch der Ladezustand der Batterie gehalten werden. In Abb. 46: ist der Ladebeginn der Batterie rechts oben, am höchsten Wert der x-Achse. Der Ladevorgang endet, wenn die x-Achse den Wert 0 erreicht. Dies erklärt, warum das Modell in Abb. 46: die Restladezeit der Batterie gegen Ende des Ladevorganges in der Nähe von 0 belässt, da die Batterie zu diesem Zeitpunkt bereits fast voll geladen ist. Anschließend ist eine Parallelverschiebung der Vorhersage des in rot markierten Ladevorganges zum Rest ersichtlich. Die gleiche Steigung lässt darauf schließen, dass dieser Bereich gut prognostiziert wird. Da in diesem Ladevorgang eine Vorklimatisierung auftrat und diese in den Trainingsdaten nicht abgebildet war, wird der Ladevorgang unzufriedenstellend vorhergesagt.

Der zweite Ladevorgang, in grün in beiden Abb. 46: und Abb. 47: dargestellt, wird zuerst gut vorhergesagt und nachdem es einen Sprung im Ladezustand sowie in der Zellspannung gab, wird der restliche Teil zwar mit gleicher Steigung prognostiziert, jedoch parallel verschoben zum Optimum. Daher kommt es auch, dass in Abb. 46: dieser Ladevorgang zuerst keine Abweichung aufweist und ab einer Restladezeit von ca. 1500 Sekunden es zur Abweichung kommt. Hintergrund dazu ist, dass in einer Datei zwei Ladevorgänge gespeichert sind. Aus diesem Grund wurde die Restladezeit falsch berechnet und daher stimmt die Prognose nicht mit dem Soll Wert der Restladezeit überein.
Der dritte Ladevorgang, in beige markiert, benötigt für eine Vollladung der Batterie startend bei ca. 62% Ladezustand ungefähr 2750 Sekunden. Der in schwarz markierte, gut prognostizierte Ladevorgang, benötigt für diese 28% (es wird real nur bis 90% Ladezustand geladen) rund 1800 Sekunden. Dies bedeutet, dass mit einer geringeren Leistung als 5kW geladen wurde. Hierbei wurde ein Ladevorgang falsch kategorisiert und sollte im Modell für Leistungen unter 5kW landen. Dies ist in den Trainingsdaten der Modellbildung für Leistungen über 5kW nicht abgebildet und konnte daher vom Modell nicht richtig prognostiziert werden.

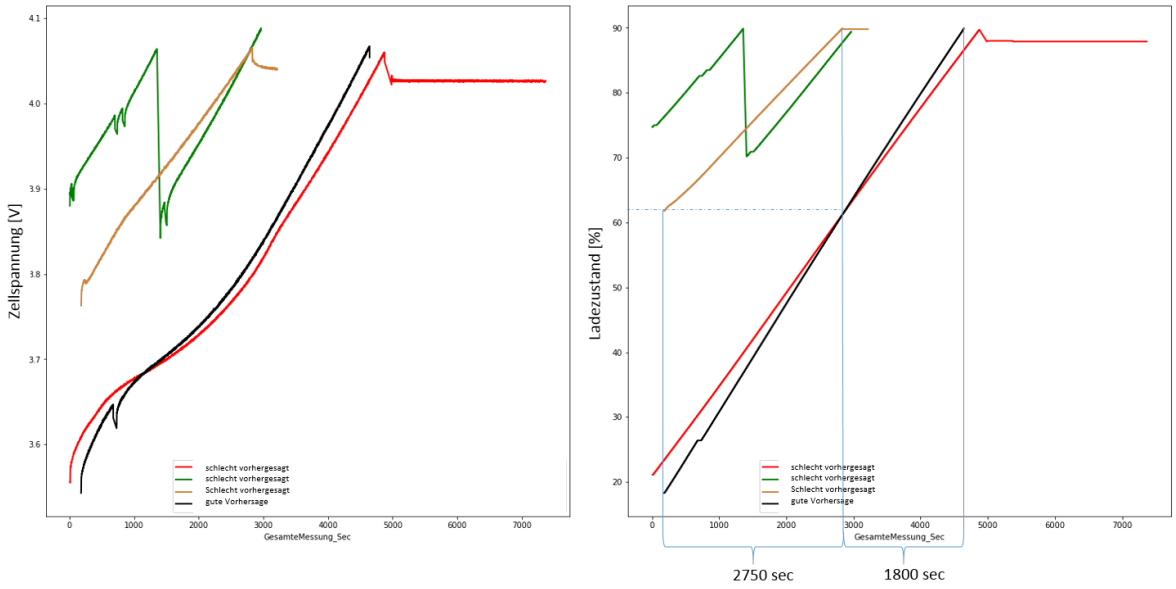

Abb. 47: Datenuntersuchung Kreuzvalidierung

Aus dieser Datenanalyse wurde ersichtlich, dass die Prognosen des Modelles anhand der Signale schlüssig erklärbar sind und dass es sich in den schlecht prognostizierten Ladevorgängen um Ausreißer handelt.

4.4.3.11 Modell 11: Kreuzvalidierung Modell 9

Die Kreuzvalidierung wurde auch auf das Modell für Ladeleistungen geringer 5kW angewandt. Die Anzahl der Ladevorgänge beträgt 242 und aus diesen entstehen 52.902 Samples mit 190 Features. Das Prinzip der Kreuzvalidierung wurde bereits in Abschnitt 3.3.4.3 erläutert. Die Anzahl der Teildatensätze wurde auf $n = 5$ gesetzt. Dabei ergibt sich wieder eine Aufteilung von 80% Trainings- zu 20% Testdaten.

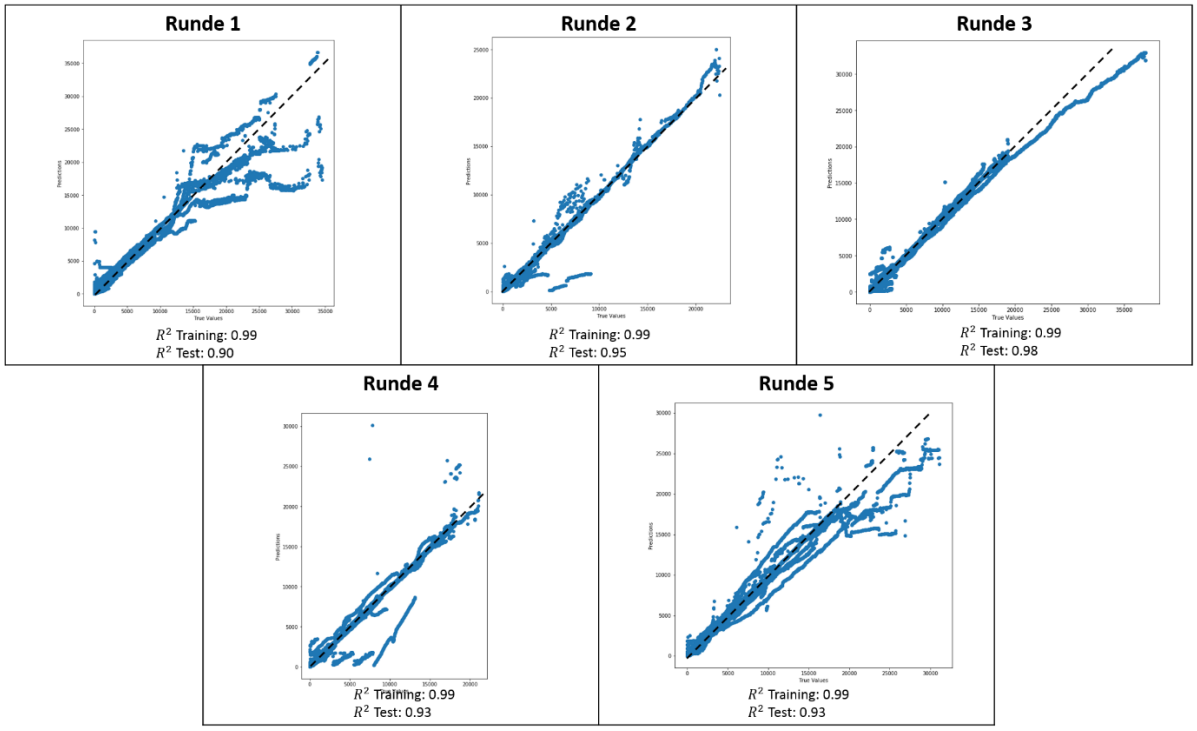

Abb. 48: Ergebnisse Kreuzvalidierung Ladeleistung kleiner 5 kW

Wie in Abb. 48: ersichtlich, kommt es in den Runden 1,4 und 5 der Kreuzvalidierung zu sehr starken Abweichungen zur schwarz strichlierten Soll – Gerade. In den Runden 2 und 3 sind die grafischen Abweichungen geringer. Der berechnete RMSE Wert aller Prognosen liegt bei 1588 was auf eine schlechte Gesamt-Prognosegüte dieses Modelles schließt. In Summe werden von den 242 Ladevorgängen, 84 Ladevorgänge zufriedenstellend prognostiziert, wie in Kapitel 4.4.2.2 beschrieben. Dies entspricht einem Anteil von 34,7%.
Wie in Abb. 26: ersichtlich, ist der Datensatz aller Ladevorgänge unter 5kW unbalanciert. Es gibt mäßig viele Ladevorgänge bei einer Leistung von ca. 4kW und 1kW, aber nur wenige in den restlichen Bereichen. Der beste Weg, dieses Problem zu beseitigen ist es, mehr Daten zu beschaffen [55].

4.4.3.12 Modell 12: Modellunterscheidung

Bei zwei Modellen, über und unter 5kW Ladeleistung muss durch einen festgelegten Vorgang zwischen den Modellen ausgewählt werden. Dies kann durch die physikalische Größe der aktuellen Leistung des Ladegerätes erfolgen. Ein weiterer Ansatz zur Bestimmung, ob die Ladeleistung unter bzw. über 5 kW liegt, kann mittels maschinellem Lernen erfolgen. Unter anderem ist dabei interessant, welche Features vom Random Forest Modell als wichtig herangezogen werden, um diese Unterscheidung zu treffen. Hierbei wird ein Supervised Learning Modell trainiert, wie in Kapitel 3.2.1 erläutert. Hierbei handelt es sich um ein Klassifikationsproblem. Es soll vorhergesagt werden, ob es sich um eine Ladeleistung größer 5kW, oder einer Ladeleistung kleiner 5kW handelt. Dazu wurde der Datensatz bearbeitet und

für jedes Sample im Trainings- und Testdatensatz der Ausgangsparameter erzeugt. Dieser Ausgangsparameter wurde „Hohe Ladeleistung“, für Leistungen größer 5kW, und „geringe Ladeleistung“, für Leistungen kleiner als 5kW, genannt. Somit kann der Random Forest Klassifikator Strukturen finden und anhand der Eingangsparameter erlernen, welche der Parameter für eine Unterteilung wichtig ist. Interessant war vor allem, welche wichtigen Eingangsparameter vom Modell für diese Entscheidung berücksichtigt werden. Die Eingangsparameter bleiben gleich wie in den Modellen 4.4.3.8 und 4.4.3.9.
In Abb. 49: wird eine Warheitsmatrix (engl.: Confusion Matrix) gezeigt, wie in Kapitel 3.3.4.2 bereits erläutert.

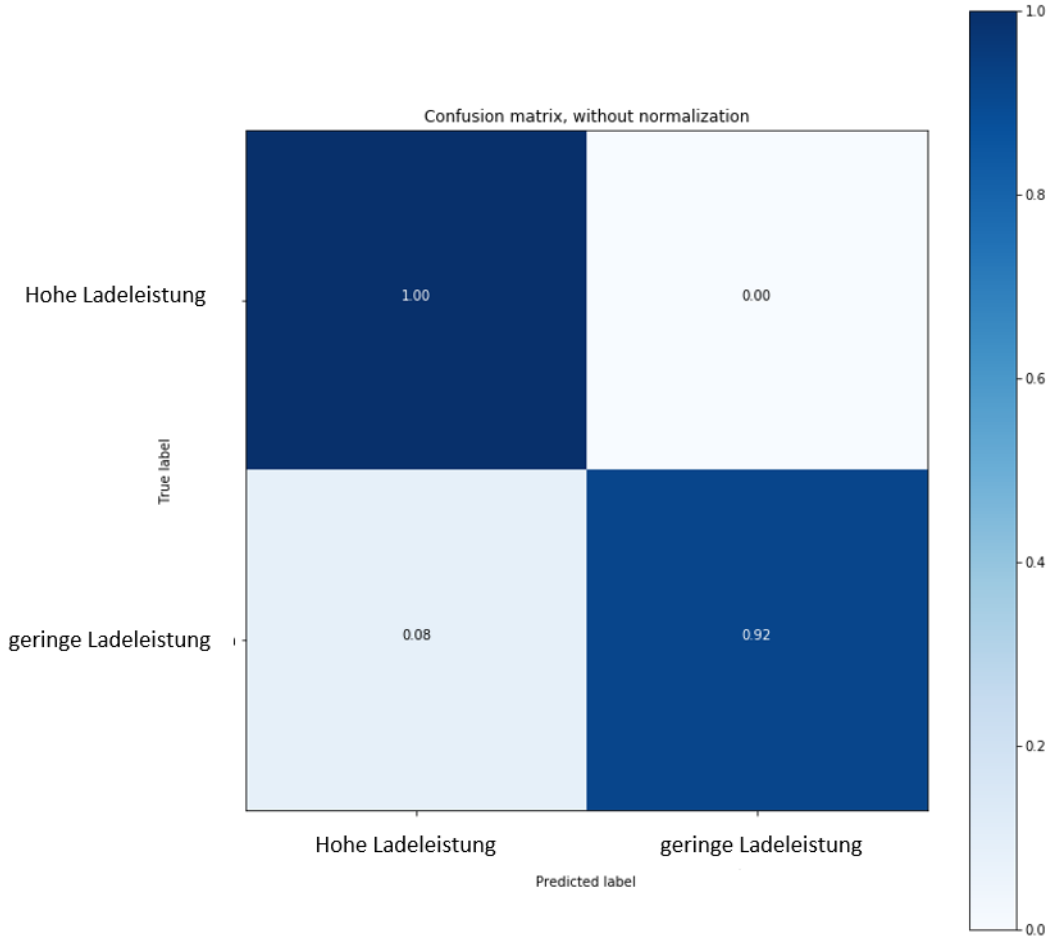

Abb. 49: Unterscheidung der Ladeleistung mittels Klassifikation

Hierbei wird ersichtlich, dass die Klasse „Hohe Ladeleistung“ in 100% der Fälle und die Klasse „geringe Ladeleistung“ in 92% der Fälle richtig vorhergesagt wurde. In 8% der Fälle wurde vom Modell eine „Hohe Ladeleistung“ vorhergesagt, richtig wäre jedoch die Klasse „geringe Ladeleistung“ gewesen.
In Abb. 50: wird die Wichtigkeit der Features dieses Random Forest Klassifikator‘s aufgezeigt. Das wichtigste Feature des Modelles ist die Ladeleistung mit einer Gewichtung von 0,29, das zweitwichtigste die Spannung des offenen Ladekreises mit einer Gewichtung von 0,19. Auf diese wurde bereits im Kapitel 4.4.3.6 eingegangen. Anschließend dienen der reale

Ladezustand und die Zellspannung als drittes und viertwichtigstes Feature mit einer Gewichtung von 0,19 und 0,07.

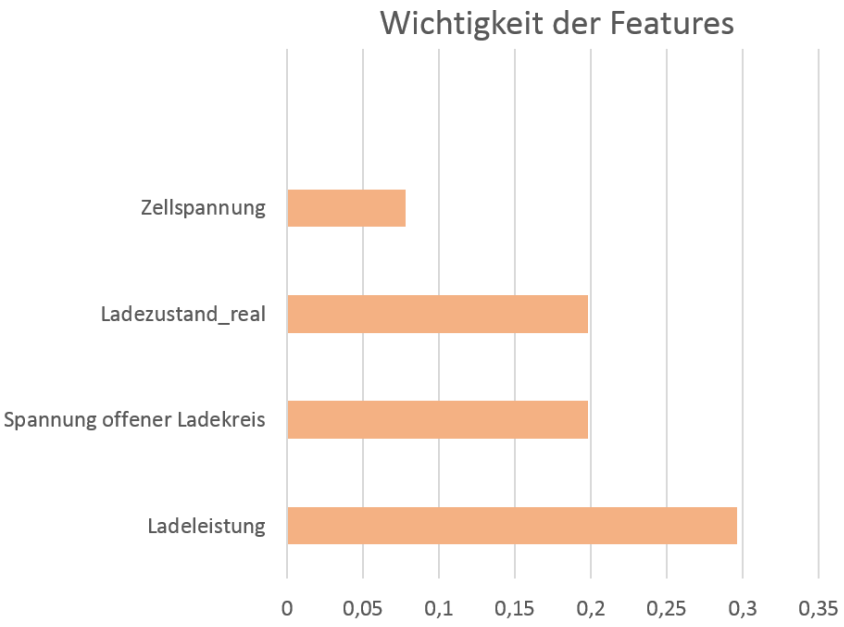

Abb. 50: Klassifikation: Wichtigkeit der Features

Aufgrund der hohen Genauigkeit, kann eine Unterteilung auch mit einem Klassifikationsmodell erfolgen.

4.4.3.13 Modell 13: Parameteroptimierung

Um die Vorhersagegüte des Modelles zu verbessern, ist es sinnvoll die Parameter des Modelles zu optimieren. Dies kann anhand der in Kapitel 3.3.5 vorgestellten Methoden erfolgen. Es wurde ein Raster an Parametern erstellt und anschließend wurde das Modell auf alle möglichen Kombinationen dieses Rasters trainiert, getestet und evaluiert. Anhand des RMSE Wertes wurde anschließend das Modell mit der besten Prognosegüte ausgewählt und somit das beste Parameterset bestimmt. Das getestete Raster an Parametern ist in Abb. 51: erläutert. Die Beschreibung der Parameter des Random Forest Modelles wurde aus [47] entnommen.

Parameter	Bereich	Beschreibung
n_estimators	[130, 180, 230]	Anzahl an Bäumen im Random Forest
max_depth	[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None]	Maximale Baumtiefe
Max_features	[auto, sqrt]	Maximale Anzahl an Features um sie an einen Baum zu testen
bootstrap	[True,False]	Kleiner Auszug von einem größeren Teil zur Bildung der Entscheidungsbäume
min_samples_leaf	[1, 2, 4]	Benötigte minimale Anzahl für einen Entscheidungsknoten
min_samples_split	[2, 5, 10]	Minimale Anzahl an Samples für internen Entscheidungsknoten

Abb. 51: Raster an Parametern, Beschreibung aus [47]

Das gezeigte Raster bietet 1296 verschiedene Kombinationen. Dieses Raster wurde mittels der maschinellen Lernexperten des Projektpartners Daimler AG abgestimmt.
Für jede einzelne Kombination wurde ein RF Modell gebildet und validiert. Das Modell mit der besten Prognosegüte wurde anhand des kleinsten RMSE Wertes ausgewählt. Dieses Modell wurde mit dem in Abb. 52: ersichtlichen Parameterset trainiert und getestet.

Bestes Parameterset:

n_estimators:	230
min_samples_split:	5
min_samples_leaf:	4
max_features:	sqrt
max_depth:	50
bootstrap:	False

Abb. 52: Ergebnis des Grid Search

Wird dieses neue Parameterset an dem Random Forest Modell angewandt, ist eine deutliche grafische Verbesserung ersichtlich, wie in Abb. 53: gezeigt. In grau ist hierbei die Prognosegüte des RF Modelles von Kapitel 4.4.3.8 abgebildet. In Blau ist die Prognosegüte des Modelles, mit den Parameterergebnissen der Rastersuche ersichtlich. Es wird deutlich, dass durch die Änderung der Parameter eine Näherung an die schwarze Soll Line möglich war und somit die Prognosegüte verbessert wurde.

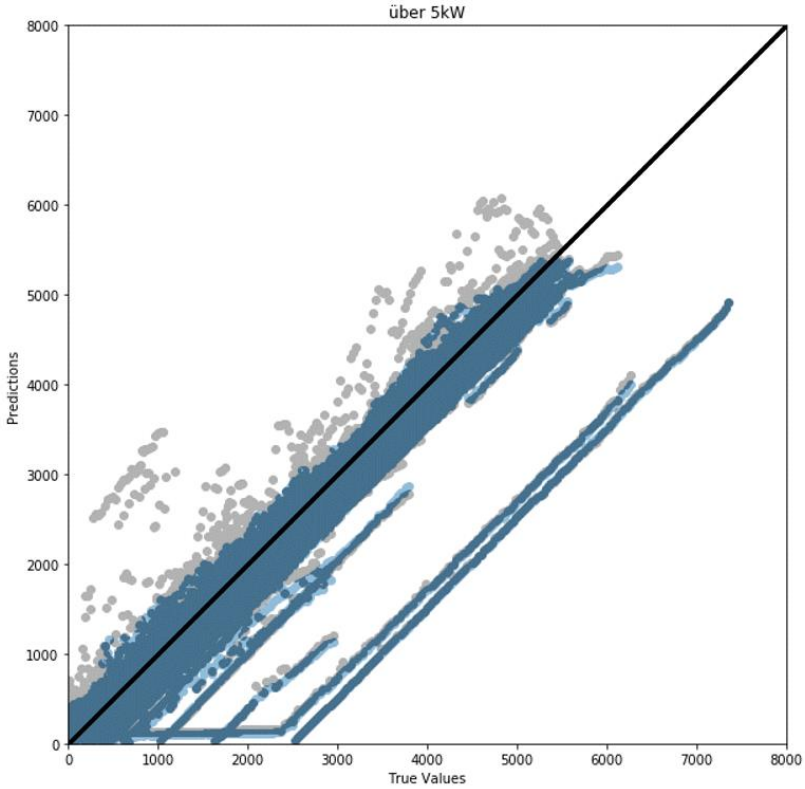

Abb. 53: Verbesserung der Prognosegüte mit neuem Parameterset – Modell > 5kW

Der RMSE Wert, bezogen auf das Modell für Ladeleistungen größer 5kW, hat sich durch die Parameteroptimierung von 120,3 auf 114,01 verbessert.

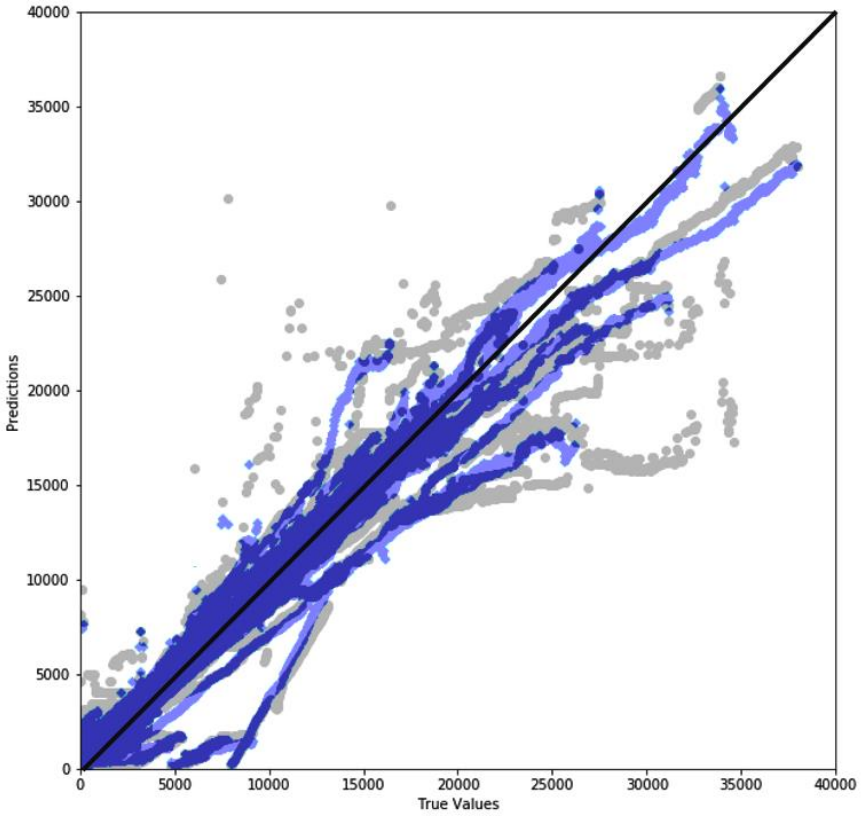

Abb. 54: Verbesserung der Prognosegüte mit neuem Parameterset – Modell < 5kW

Das gleiche Parameterset wurde auch auf das Modell für Ladeleistungen kleiner als 5kW angewendet. In grau ist hierbei die Prognosegüte des RF Modelles von Kapitel 4.4.3.9 abgebildet. In Blau ist die Prognosegüte des Modelles, mit den Parameterergebnissen der Rastersuche ersichtlich. Der RMSE Wert hat sich durch die Parameteroptimierung von 1588 auf 1376 verbessert.

5 Interpretation und Diskussion der Ergebnisse

Wie in Kapitel 4.4.3 erwähnt, wurden schrittweise komplexere Modelle verwendet, mehr Daten hinzugefügt, sowie schlussendlich die Parameter optimiert. Dieser iterative Prozess erzeugte Stück für Stück eine verbesserte Prognosegüte.
Wie in Kapitel 4.3.7, erwähnt, wurden seitens Daimler AG kontinuierlich neue Daten zur Verfügung gestellt.
Diese Iterationsschritte wurden in Abb. 31: bereits erläutert. In Abb. 55: werden hierbei zusätzlich die Ergebnisse und Erkenntnisse jedes einzelnen Iterationsschrittes aufgeführt. Diese werden im Folgenden erläutert.

Modell Nr.	Datenanzahl	Ziel	Ergebnis
1	101	Bestimmen des besten Ansatzes	Zeitfenster mit beschreibender Statistik liefert die beste Prognosegüte
2	101	Bestimmen des besten Modelles	Random Forest liefert eine bessere Prognosegüte als LLR
3	536	Eignung dieses Anwendungsfalles für maschinelles Lernen	Ja, maschinelles Lernen ist geeignet
4	3131	Anwendbarkeit Modell 3 auf restliche Datensätze	Kein gutes Ergebnis, erneute Überprüfung mittels 80% / 20% Aufteilung
5	3131	Bestätigung der getroffenen Aussagen von Modell 4	Erkenntnisse von Modell 4 bestätigt
6	3023	Sample Anzahl möglichst hoch halten, Prognosegüte verbessern	Große Abweichungen in der Prognose der Restladezeit
7	108	Sample Anzahl möglichst hoch halten, Prognosegüte verbessern	Große Abweichungen in der Prognose der Restladezeit
8	2067	Prognosegüte verbessern	Gute Prognosegüte
9	242	Prognosegüte verbessern	Gute Prognosegüte
10	2067	Bewertung der Modelle bzgl. des ganzen Datensatzes	Prognosegüte konstant über den ganzen Datensatz gut
11	242	Bewertung der Modelle bzgl. des ganzen Datensatzes	Prognosegüte konstant über den ganzen Datensatz gut
12	2309	Modellunterscheidung mittels ML	physikalisch anhand von Signalen oder mittels ML Algorithmus möglich
13	2544	Parameteroptimierung	Verbesserung der Prognosegüte

Abb. 55: Ergebnisse der Iterationsschritte

Das Ergebnis der ersten Modellbildung zeigte, das der Ansatz eines Zeitfensters mit beschreibender Statistik, wie in Kapitel 4.3.3 beschrieben, die beste Prognosegüte liefert.
In der zweiten Modellbildung wurde die Erkenntnis gewonnen, dass die Prognosegüte des RF Modelles besser ist, als des LLR Modelles.
Der dritte Iterationsschritt zeigte, dass maschinelles Lernen für diesen Anwendungsfall geeignet ist.
Im vierten Iterationsschritt wurde getestet, ob Modell Nr. 3 auf den ganzen Datensatz anwendbar ist und welche Abweichungen dadurch in den Daten erkennbar sind. Hierbei wurde festgestellt, dass sehr unterschiedliche Ladezeiten benötigt werden um die Batterie um 10% Ladezustand zu laden. Diese Erkenntnisse wurden anhand einer untypischen Aufteilung von 20% Trainings- zu 80% Testdaten gebildet.
Daher wurde im Modell 5 diese Aussage mit einer typischen Aufteilung, wie in Kapitel 3.3.2.4 beschrieben, überprüft. Hierbei wurde festgestellt, dass eine Unterteilung der Daten sinnvoll und notwendig ist, um die Prognosegüte zu erhöhen.
In Modell 6 und 7 wurden die Daten anhand der Ladedauer unterteilt. Ziel war es, die Sample Anzahl möglichst hoch zu halten, da bei dieser Unterteilung keine Daten entfernt werden müssen. Dies führte zum Ergebnis, dass große Abweichungen von beiden Modellen prognostiziert wurden.

In Modell 8 und 9 wurden die Daten anhand der Ladeleistung unterteilt. Um diese Unterteilung machen zu können, mussten 822 Ladevorgänge entfernt werden. Die Prognosegüte von Modell 8 war anhand der Trainings- und Testdaten sehr gut. In Modell 9 kam es zu deutlichen Abweichungen, besonders bei sehr geringen Ladeleistungen. Diese Abweichungen sind auf die geringe Datenanzahl zurückzuführen. Die Kombination der beiden Modelle 8 und 9 liefert die beste, bisher erreichte Genauigkeit.
Das Ergebnis der Kreuzvalidierung in Modell 10 und 11 zeigt, dass die Prognosegüte über den ganzen Datensatz sehr ähnlich ist und somit die Generalisierungsfähigkeit der Modelle 8 und 9 beweist. Weiters wurde in der Analyse der Ergebnisse ersichtlich, dass die Modellprognose schlüssig erklärbar ist und dass es sich in den schlecht prognostizierten Ladevorgängen um Ausreißer handelt.
In Modellbildung 12 wurde die Erkenntnis gewonnen, dass eine Unterscheidung der Ladeleistung sowohl physikalisch anhand der Signale, als auch durch ein RF Klassifikationsmodell erfolgen kann.
Im letzten Schritt, wurde die Prognosegüte durch eine Parameteroptimierung beider Modelle 8 und 9 verbessert.

Im Folgenden werden nun die wichtigsten Iterationsschritte von Abb. 55: ausgewählt. Hierbei wurden nur Iterationsschritte aus dem Abschnitt 4.4.3 herangezogen, die zu einer Verbesserung der Prognosegüte führten. Diese schrittweise Verbesserung wird in Abb. 56: dargestellt. Die Iteration gliedert sich nun in 7 Schritte die im Folgenden erläutert werden.
In orange wird hier die Anzahl der Daten, sprich die Anzahl der Ladevorgänge, abgebildet. In grün ist die Bewertung der Prognose abgebildet, wie in Kapitel 4.4.2.1 beschrieben. Hierbei wird die Anzahl der Prognosen, die das Bewertungskriterium laut Abschnitt 4.4.2.1 erfüllen, auf die endgültige Datenanzahl von 2544 bezogen.

Die erste Iteration beginnt mit 101 Dateien als Trainings- sowie Testdatensatz, wie in Abschnitt 4.4.3.1 beschrieben. Aufgrund der geringen Anzahl an Daten lag die Anzahl der Prognosen, die das Bewertungskriterium erfüllen bei ca. 4%.
Im zweiten Iterationsschritt wurden die Daten auf 536 Stück erhöht, wie in Kapitel 4.4.3.3 erläutert. Die Prognosegüte erhöhte sich auf 21%.
Im dritten Iterationsschritt wurden zwei verschiedene Modelle gebildet, wie in Abschnitt 4.4.3.8 und 4.4.3.9, beschrieben. Die Prognosegüte der beiden Modelle wird hierbei summiert und zu einer Prognosegüte zusammengefasst. Die Datenbasis erhöhte sich auf 2309 Ladevorgänge. Dabei wurden 2067 Ladevorgänge zum Bilden des Modelles größer 5kW und 242 Ladevorgänge für das Bilden des Modelles kleiner 5kW verwendet. Somit verbesserte sich die Prognosegüte auf knapp 40,8%.
Die Datenbasis blieb vom dritten zum vierten Iterationsschritt gleich, jedoch wurde ein Fehler in der Datenaufbereitung ausgebessert. Die Daten wurden getrennt voneinander in verschiedenen Bereichen skaliert. Daher ist eine Verbesserung der Prognosegüte auf 75,26% ohne Erhöhung der Datengrundlage sichtbar.
Auch im 5. Iterationsschritt blieb die Datenbasis gleich. Hier wurde das gefundene Parameterset von Abb. 52: verwendet. Dies bewirkte eine Verbesserung der Prognosegüte auf 76,97%.

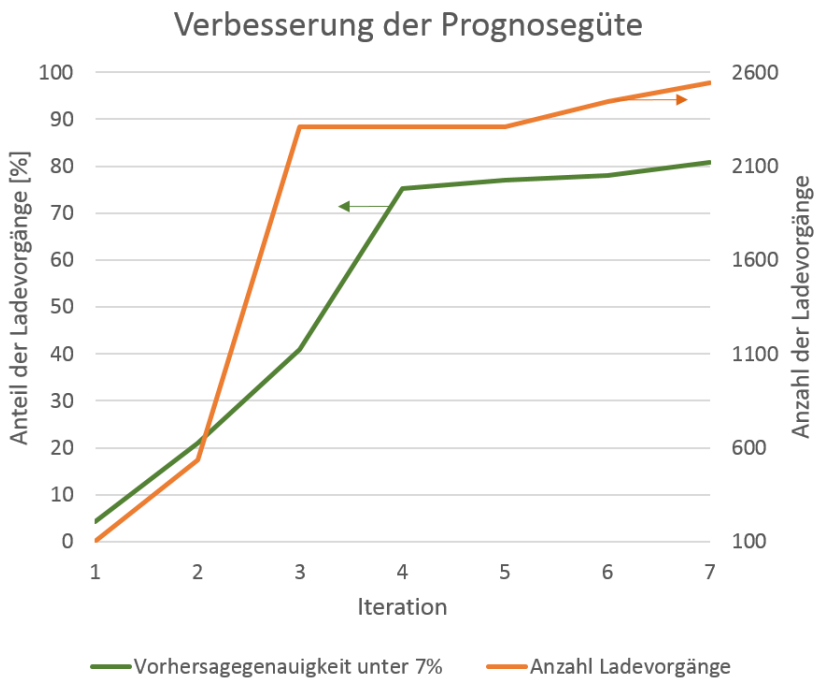

Abb. 56: Verbesserung der Prognosegüte – hohe Ladeleistung

Die letzten beiden Iterationsschritte wurden in Kapitel 4.4.3 der Modellbildung nicht erwähnt, da hierbei nur Daten hinzugefügt wurden. Die Modellbildung wurde wie bisher mit den beiden Modellen größer/kleiner 5kW, mittels der in Abschnitt 4.4.3.13 gefundenen Parameter ausgeführt. Im 6. Schritt wurden die Daten um 136 Ladevorgänge erweitert und dies bewirkte eine Verbesserung der Prognosegüte auf 78,07%. Im letzten Iterationsschritt wurde die Datenanzahl um 99 Ladevorgänge erhöht und dies bewirkte eine Verbesserung der Prognosegüte auf 80,72%.
Vergleicht man diese Prognosegüte nun mit dem bisherigen Modell, wie in Kapitel 4.4.2.1 beschrieben, wird eine Prognosegüte des Modelles größer 5kW von 83,3% (1896 Ladevorgänge) und des Modelles kleiner 5kW von 59,8% (158 Ladevorgänge) erreicht. Dies ergibt eine Summe von 2054 Ladevorgängen die laut dem in 4.4.2.1 erwähnten Bewertungskriterium zufriedenstellend prognostiziert werden. Wie in Abb. 18: erwähnt, sind dies im bisherigen physikalischen Prognosemodell nur 536 Stück. In Abb. 57: wird der direkte Vergleich beider Modelle abgebildet. Aufgrund der in Kapitel 4.3.2 beschriebenen Maßnahme, die Daten zu reduzieren, da grundlegend wichtige Signale fehlen und der oben beschriebenen Iterationsschritte 6 und 7, in denen Daten hinzugefügt wurden, bezieht sich die Datengrundlage nun auf 2544 Ladevorgänge.

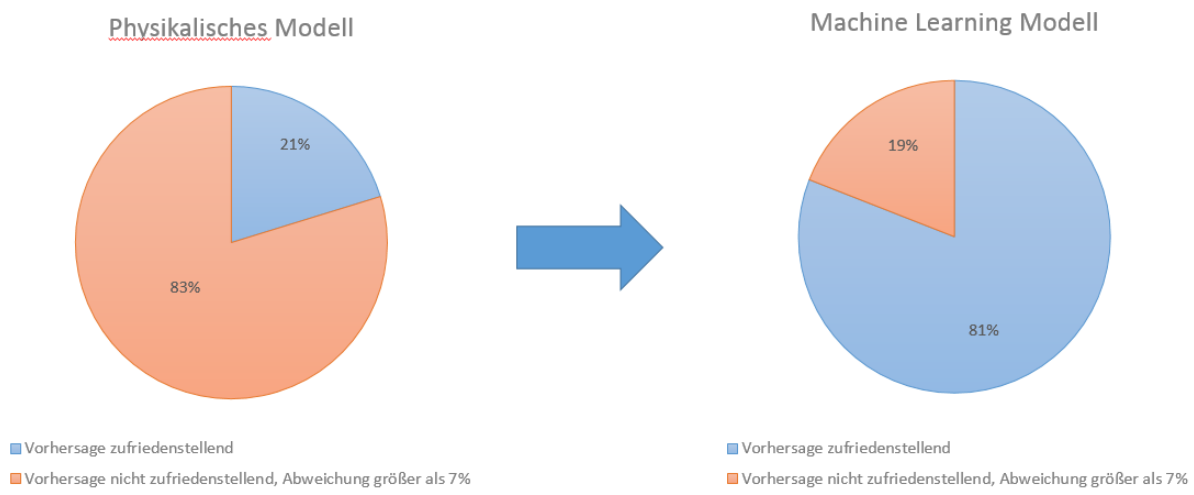

Abb. 57: Vergleich Maschinelles Lernmodell zu bisherigem Modell

Bezieht man nun das physikalische Modell und das gebildete maschinelle Lernmodell auf diese Anzahl von 2544 Ladevorgängen, erfüllt die Prognose des physikalischen Modells das Bewertungskriterium zu 21% und die Prognose des maschinellen Lernmodelles zu 81%. Somit ist durch die Kombination beider in 4.4.3.8 und 4.4.3.9 erwähnten maschinellen Lernmodelle eine Verbesserung der Prognosegüte etwa um den **Faktor 4** möglich.

6 Zusammenfassung und Handlungsempfehlungen

Die erarbeiteten Ergebnisse der Masterarbeit werden nachfolgend zusammengefasst und Handlungsempfehlungen bezüglich der weiteren Entwicklung des Machine Learning Anwendungsfalles gegeben.

6.1 Zusammenfassung

In dieser Arbeit wurden die Grundlagen des Batterieladevorganges sowie der aktuelle Stand der Technik bezüglich der Prognose der Restladezeit der Batterie erläutert. Es wird ein Überblick über die Grundlagen des maschinellen Lernens gegeben und ein Vorgehensmodell zur Modellbildung, beginnend mit der Datenanalyse bis zur Modellevaluation mit allen Teilschritten erklärt. Anschließend wird der Anwendungsfall, die Prognose der Restladezeit der Batterie erklärt und erläutert. Ziel ist es, mittels einem maschinellem Lernmodell eine Verbesserung der Prognosegüte der Restladezeit der Batterie zu erreichen.
Es wird auf den aktuellen Stand der Technik der Hybridfahrzeuge eingegangen und erklärt wie die Daten in den 16 verschiedenen Testfahrzeugen generiert werden. Die verwendbare Datengrundlage umfasst 3366 Ladevorgänge aus Hybridfahrzeugen der E-Klasse und S-Klasse des Projektpartners Daimler AG der Marke Mercedes-Benz. Diese vorliegenden Daten wurden analysiert und auf Qualität geprüft. Diese Daten wurden anschließend bearbeitet und für maschinelles Lernen vorbereitet. Hierbei wurden die Daten aufbereitet indem fehlende Werte interpoliert, auf den nötigen Zeitbereich gefiltert und die Restladezeit zu jedem Zeitpunkt berechnet wurde. Anschließend wurden drei verschiedene Ansätze untersucht, um Zeitreihen in ein maschinelles Lernproblem überzuführen. Es wird beschrieben wie die einzelnen Daten eingelesen, anhand der Ladeleistung unterteilt und skaliert wurden. Zuerst wurde die grundlegende Fragestellung, ob eine Prognose der Restladezeit mittels maschinellem Lernen möglich ist mit Ja beantwortet, indem ein vereinfachtes Modell gebildet wurde. Es wurden Schritt für Schritt mehr Daten hinzugefügt und somit die Erkenntnis gewonnen, dass aufgrund des unbalancierten Datensatzes und der geringen Anzahl an Daten, zwei verschiedene Modelle zur Prognose der Restladezeit der Batterie eine bessere Lösung sind. Es wurde ein Modell für Ladeleistungen größer 5 kW und ein weiteres für Ladeleistungen kleiner 5kW gebildet. Die Kombination dieser beiden Modelle lieferte die beste Genauigkeit und somit wurden 81% der verwendeten Ladevorgänge zufriedenstellend prognostiziert. Die bisherige, des Projektpartners Daimler AG implementierte Prognose, lieferte eine Genauigkeit von 21% und somit wurde eine Verbesserung um den Faktor 4 mittels der gebildeten maschinellen Lernmodelle erreicht.

6.2 Handlungsempfehlungen

In diesem Abschnitt werden Verbesserungen hinsichtlich der Datengrundlage und der Schritte zur weiteren Vorgehensweise aufgezeigt.
Die Datengrundlage soll bei neuerlicher Modellbildung deutlich vergrößert werden.

Es ist wichtig darauf zu achten, dass die Datenaufzeichnung einheitlich ist und alle batteriebezogenen Signale aufgezeichnet werden.
Diese Datengrundlage soll unterschiedlichste Bedingungen abbilden. Es sollen verschiedenste Temperaturebereiche sowie Lastzustände abgebildet sein. In diesen Lastzuständen sollen Verbraucher wie Klimaanlage und Heizung in kompletter Bandbreite von 0% bis zur Volllast enthalten sein. Die Daten sollen von verschiedenen Baureihen mit unterschiedlicher Batterietechnologie und Batteriekapazität stammen. Es soll das ganze Ladeleistungsspektrum von 0 bis 150 kW abgedeckt sein, mittels Gleichstrom und Wechselstromladen.
Weiters sollen Kenngrößen der Batterie, die Auskunft über das Alter, den Zustand, der aktuellen Kapazität und der Anzahl der bereits geladenen Ladezyklen geben, in den Daten enthalten sein.
Zusätzlich soll die Performance anderer maschineller Lernmethoden getestet und evaluiert werden. Zum Beispiel mittels Support-Vektor Maschinen und Neuronaler Netzen. Dabei soll ein Fokus auf das Feature Engineering gelegt werden. Diese neuen Features sollen mittels Batterie- und Fachexperten des maschinellen Lernens generiert werden.
Es soll ein mitlernendes System in Betracht gezogen werden. Dies macht aus vielen Gründen Sinn. Die verschiedenen Nutzerverhalten mit unterschiedlichen infrastrukturellen Gegebenheiten führen zu ungleichen Ladekurven. Die Dauer des Ladevorganges wird stark durch die vorliegende Umgebungstemperatur und den Vorlieben des Nutzers beeinflusst. Darunter zählt zum Beispiel die Vorliebe in ein vorgeheiztes Fahrzeug zu steigen, was die Ladedauer, je nach aktueller Ladeleistung, stark beeinflussen kann. Weiters verändert sich die Alterung und der „Gesundheitszustand“ der Batterie je nach Nutzerverhalten anders. Ladungen an einer Schnellladesäule führen zu einer schnelleren Alterung der Batterie, als Ladungen mit geringer Leistung. Somit ändert sich die Ladedauer der Batterie mit der Zeit, je nach Ladeverhalten des Nutzers. Abhilfe könnte daher eine personalisierte, mitlernende Ladeprognose schaffen, die das Nutzerverhalten erlernt und somit auch Wissen über den „Gesundheitszustand“ der Batterie aufbaut. Somit könnte eine langfristige und gesteigerte Prognosegenauigkeit erreicht werden.

7 Literaturverzeichnis

[1] Deutsches Bundesministerium für Wirtschaft und Energie (bmwi.de): https://www.bmwi.de/Redaktion/DE/Dossier/elektromobilitaet.html, Zugriff am 07.01.2019

[2] Anton Karle: *Elektromobilität – Grundlagen und Praxis*, Carl Hanser Verlag, Seiten 88 - 103, 2015

[3] Jun Bi, Yongxing Wang, Shuai Sun, Wei Guan: *Predicting Charging Time of Battery Electric Vehicles Based on Regression and Time-Series Methods: A Case Study of Beijing*, Energies 2018/11, 2018

[4] Peter Kurzweil, Otto K. Dietlmeier.: *Elektrochemische Speicher*, Springer Verlag, 2015

[5] Robert A. Meyers.: *Encyclopedia of Sustainability Science and Technology*, Seite 684-686, 2012

[6] Standardization for Automotive Development (asam.net): https://www.asam.net/standards/detail/ods/wiki/ , Zugriff am 18.10.2018

[7] Standardization for Automotive Development (asam.net) https://www.asam.net/standards/detail/mdf/ , Zugriff am 05.01.2019

[8] Daniel Ch. von Grünigen: *Digitale Signalverarbeitung*, Carl Hanser Verlag, 2014

[9] Associtation for Standardisation of Automation and Measuring Systems (pypi.org): https://pypi.org/project/asammdf/ , Zugriff am 19.12.2018

[10] E Marie-Aude Aufaure; *Business Intelligence*, Springer Verlag, 2012

[11] Sebastian Raschka: *Python Machine Learning – Unlock deeper insights into machine learning with this vital guide to cutting-edge predictive analytics,* Packt Publishing, 2016

[12] Sunil L. Kukreja , Johan Löfberg, Martin J. Brenner: A *least absolute shrinkage and selection operator (Lasso) for nonlinear system identification,* International Forum on Aeroelasticity and Structural Dynamics, 2006

[13] Robert Tibshirani: *Regression shrinkage and selection via the lasso,* Journal of the Royal Statistical Society, 1996

[14] Leo Breimann: *Bagging Predictors,* Department of Statistics - University of California Berkeley, 1994

[15] Marina Skurichina and Robert P. W. Duin: *Bagging, Boosting and the Random Subspace Method for Linear Classifiers,* Springer-Verlag London Limited, 2002

[16] Gilles Louppe: *Understanding Random Forests: From Theory to Practice,* University of Liège, 2014

[17] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou und 9 weitere: *Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm,* arXiv:1712.01815, 2017

[18] Csaba Szepesvári: *Algorithms for Reinforcement Learning*, Morgan & Claypool Publishers, 2009

[19] David Chicco: *Ten quick tips for machine learning in computational biology,* BioData Mining, 2017

[20] Colin Shearer: *The CRISP-DM Model: The New Blueprint for Data,* Journal of Data Warehousing, 2000

[21] Machine Learning Mastery by PHD Jason Brownlee (machinelearningmastery.com): https://machinelearningmastery.com/understand-machine-learning-data-descriptive-statistics-python/, Zugriff am 30.01.2019

[22] Dr. Siegfried Vössner: *Vorlesungsunterlagen - Optimization Methods for Operations Planning*, Technische Universität Graz – Maschinenbau und Betriebsinformatik, 2018

[23] Charu C. Aggarwal.: *Outlier Analysis*, Springer Vieweg Verlag, 2013

[24] Hans-Peter Kriegel, Peer Kröger, Arthur Zimek: *Outlier Detection Techniques*, The 2010 SIAM International Conference on Data Mining, 2010

[25] Manohar Swamynathan: *Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics Using Python*, Springer Verlag, 2017

[26] Chi Nhan Nguyen, Oliver Zeigermann: *Machine Learning: kurz & gut*, O'Reilly Verlag, 2018

[27] Dunteman George H.: *Principal Components Analysis*, SAGE Publications, 1989

[28] Huan Liu, Hiroshi Motoda: *Feature Selection: For Knowledge Discovery and Data Mining*, Springer Science and Business Media, 1998

[29] Andrew Ng: *Machine Learning and AI via Brain Simulations,* Stanford University, 2017

[30] James Max Kanter, Kalyan Veeramachaneni: *Deep Feature Synthesis: Towards Automating Data Science Endeavors,* MIT, 2015

[31] Dipanjan Sarkar, Raghav Bali, Tushar Sharma: *Practical Machine Learning with Python: A Problem-Solver's Guide to Building Real-World Intelligent Systems,* Springer, 2018

[32] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani: *An Indtroduction to Statistical Learning,* Springer, 2013

[33] scikit-learn Machine Learning in Python (scikit-learn.org): https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html, Zugriff am 02.01.2019

[34] Trevor Hastie, Robert Tibshirani, Jerome Friedman: *The Elements of Statistical Learning*, Springer, 2017

[35] Wolfgang Ertel: *Grundkurs Künstliche Intelligenz 4. Auflage*, Springer Vieweg Verlag, Seite 2, 2016

[36] Rajesh Khada: https://towardsdatascience.com/machine-learning-types-2-c1291d4f04b1, Zugriff am 04.01.2019

[37] Jörg Frochte: *Maschinelles Lernen, Grundlagen und Algorithmen in Python*, Carl Hanser Verlag, 2018

[38] Brian Ripley: *Pattern Recognition and Neural Networks*, 1996

[39] Andreas C. Müller und Sarah Guido: *Einführung in Machine Learning mit Python*, O'Reilly Verlag, 2017

[40] Kelly H. Zuo, Kemal Tuncali, Stuart G. Silverman: *Correlation and Simple Linear Regression,* Radiology, 2003

[41] Leo Breimann: *Random Forests*, Statistics Department - University of California Berkeley , 2001

[42] Misha Denil , David Matheson, Nando de Freitas: Narrowing the Gap: *Random Forests In Theory and In Practice*, W&CP volume 32, 2014

[43] Sonia Singh, Priyanka Gupta: *Comparative Study ID3, CART and C4.5 Decision tree Algorithm*, International Journal of Advanced Information Science and Technology No.27, 2014

[44] Andy Liaw and Matthew Wiener: Classification and Regression by Random Forest, R News Vol/3, 2002

[45] scikit-learn Machine Learning in Python (scikit-learn.org): https://scikit-learn.org/stable/modules/linear_model.html#lasso, Zugriff am 16.01.2019

[46] scikit-learn Machine Learning in Python (scikit-learn.org): https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html, Zugriff am 16.01.2019

[47] scikit-learn Machine Learning in Python (scikit-learn.org): https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html ,Zugriff am 16.01.2019

[48] Thais Mayumi Oshiro, Pedro Santoro Perez, and Jos´e Augusto Baranauskas: *How Many Trees in a Random Forest?*, Springer-Verlag Berlin Heidelberg, 2012

[49] E.Rich. *Artificial Intelligence*. McGraw-Hill, 1983.

[50] A. G. Asuero, A. Sayago, A. G. Gonzalez: *The Correlation Coefficient: An Overview,* Chritical Reviews in Analytical Chemistry, 2006

[51] Ludwig Fahrmeir, Thomas Kneib, Stefan Lang: *Regression – Modelle, Methoden und Anwendungen*, 2. Auflage, 2009

[52] Tom Fawcett: *An introduction to ROC analysis*, Pattern Recognition Letters 27, 2006

[53] Claude Sammut, Geoffrey I. Webb: *Encyclopedia of Machine Learning*, Springer, 2011

[54] Rob J. Hyndman, Anne B. Koehler: *Another look at measures of forecast accuracy,* International Journal of Forecasting 22, 2006

[55] Davide Chicco: *Ten quick tips for machine learning in a computational biology,* Bio Med Central, 2017

[56] Pedro Domingos: *A Few Useful Things to Know about Machine Learning*, 2017

8 Abbildungsverzeichnis

Anhang

Signalliste

Bezeichnung	Bitanzahl	Physikalischer Bereich
Außenlufttemperatur	8	-40..+85°C
Modus On Board Lader	3	0 Off; 1 Buck, 2 Boost 3 Locked, 4 Lhom; 5 ZKE
Ist Transformatortemperatur DC/DC Wandlerplatine	8	-50...+204
DC/DC Strom	14	-819,2…+819A
DC/DC Spannung	9	0…51V
DC/Dc Last	8	0…100%
DC/DC Strom V2	14	-819,2…+819A
DC/DC Spannung V2	9	0…51V
Ist-Modultemperatur DC/DC-Wandler	8	-50..+204 °C
Datum/Zeit Tag	7	1..31 Days
Dateiname	-	-
Strom für Klimaanlage	8	0..127A
Spannung für Klimaanlage	8	0..819V
Ladezustand Steuergerät	10	0..100%
Zeit	-	sec
Identifikationsnummer	-	-
Strom Ladegerät AC	11	-100..+100A
maximal möglicher Ladestrom AC	7	0..63A
Anzahl der AC Eingangsphasen	2	0 Eine Phase 1 Zwei Phasen 2 drei Phasen
Anzahl der genutzten AC Eingangsphasen	2	0 Eine Phase 1 Zwei Phasen 2 drei Phasen
aktuelle AC Leistung des Ladegerätes	8	0..25,4kW
Spannung Ladegerät DC	9	0..510V
Zustand Ladesteckerklappe	2	0 Offen, 1 geschlossen, 2 nicht definiert
Kühlflächentemperatur Ladegerät	8	-40..+214°C
Kupplungstemperatur Ladegerät	8	-40..+214°C
Status Ladekontrolle	3	0 verbunden 1 digitale Kommunikation nötig 2
Strom Ladegerät DC	12	-200..+200A
maximal zur Verfügung stehender DC Strom	10	0..60A
maximal verfügbare Gleichstrom Leistung	8	0..25,4kW
Spannung Ladegerät DC	13	0…819V

Temperatur Wandlerplatine Ladegerät	8	-40..+214°C
Fehler Ladegerät	2	0 nicht definiert 1 Fehler 2 kein Fehler
Status Ladegerät	3	0 Low Power Mode 1 Standby 2 Ready to Charge 3 Charging
Ladegerät Proximity	2	0 keine Proximity 1 Proximity erkannt 2 S3 opened in Charger coupler plug
Stromanforderung Ladegerät	11	0..204,6A
Spannungsanforderung Ladegerät	13	0..819V
Anforderung Modus Ladegerät	2	0 Standby, 1 Niedrigleistungsmodus, 2 Laden
Kilometerstand	24	0..999999,6 km
14 V Batterie Istspannung	8	0..25,4V
Status Batterietrennschalter Hochspannungsbordnetz	2	0 Schalter offen 1 Schalter in Zwischenposition 2 Schalter geschlossen
momentan erlaubte höchste Zellenladespannung Batterie	12	1,5..4,5V
momentan erlaubte niedrigste Zellenladespannung Batterie	12	1,5..4,5V
maximale Zellenspannung Batterie	12	1,5..4,5V
minimale Zellenspannung Batterie	12	1,5..4,5V
Ist-Strom Batterie Hybrid	14	-819,2..+819,2A
Ist-Strom Batterie Elektrofahrzeug	14	-819,2..+819,2A
Status Batterie voll geladen	2	0 None 1 SOC Kunde erreicht 2 Batterie voll
Innenwiderstand Batterie beim Laden	16	0..65534 mOhm
Innenwiderstand Batterie beim Entladen	16	0..65534 mOhm
Isolationsmessung Batterie zugelassen	2	0 Isolationsmessung deaktiviert 1 Isolationsmessung aktiv 2 Isolationsmessung teilweise aktiv
Isolierwiderstandswert Batterie	16	0..65534 Ohm/V
Isolationsspannung Batterie zwischen HV+ und Masse	13	0..819 V
Referenzkapazität Batterie für SOC-Berechnung	16	0..6553400 As
Batterie Innen-Ladewiderstand	16	0..65.534 Ohm
Batterie Innen-Entladewiderstand	16	0..65.534 Ohm
Ist Ladezustand Batterie	10	0..100 %

aktueller minimaler Ladezustand Batterie	10	0..100 %
aktueller maximaler Ladezustand Batterie	10	0..100 %
Batterietemperatur	8	-50..+204 °C
maximale Batterietemperatur	8	-40..+160 °C
minimale Batterietemperatur	8	-40..+160 °C
Ist Spannung Batterie	13	0..819V
Aktuelle Leerlaufspannung Batterie bezogen auf aktuellen SOC	13	0..819V
Batterieladezustand Display (Kunde)	10	0..100%
Strom Heizung	8	0..127A
Spannung Heizung	8	0..819V
Restladezeit	-	sec
Dateiname des Ladevorganges	-	-