Modellieren vom Wärmebedarf eines Gebäudes mittels Machine-Learning

Studiengang Verfahrens und Umwelttechnik
in der Fakultät Maschinenbau

Bachelorarbeit

zur Erlangung des akademischen Grades
Bachelor of Engineering

vorgelegt von

Patrick Wrazidlo
geboren am 05.05.1999

im August 2025

Erstprüfer:	Prof. Dr.-Ing. R. Erpelding
Zweitprüfer:	Dr.-Ing. R. Wittenburg

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Abschlussarbeit selbstständig und nur unter Verwendung der von mir angegebenen Quellen und Hilfsmittel verfasst zu haben. Sowohl inhaltlich als auch wörtlich entnommene Inhalte wurden als solche kenntlich gemacht. Die Arbeit hat in dieser oder vergleichbarer Form noch keinem anderem Prüfungsgremium vorgelegen.

Datum: 27.08.2025 Unterschrift: P. Wacide

Danksagung

Diese Bachelorarbeit markiert den Abschluss eines wichtigen Lebensabschnitts. Ihr Gelingen wäre ohne die Unterstützung zahlreicher Menschen nicht möglich gewesen, bei denen ich mich an dieser Stelle von Herzen bedanken möchte.
Mein tiefster Dank gilt der Firma Theta Concepts. Sie hat mir die außergewöhnliche Chance gegeben, mich in ein für mich neues Themenfeld einzuarbeiten und es mit meinen ingenieurwissenschaftlichen Kenntnissen zu verbinden. Damit wurde mir der Weg in einen faszinierenden Arbeitsbereich eröffnet, in dem ich meine berufliche Zukunft sehe. Ein besonderer Dank gebührt dabei meinem Zweitprüfer und betrieblichen Betreuer, Herrn Dr.-Ing. Wittenburg, für die exzellente Begleitung während meines Praktikums und dieser Abschlussarbeit.
Ebenso herzlich danke ich meinem Erstprüfer, Herrn Prof. Dr.-Ing. Erpelding. Seine vertrauensvolle Betreuung und die Freiheit, mein Thema eigenständig zu gestalten, haben maßgeblich zum Erfolg dieser Arbeit beigetragen. Seine unterstützende und offene Art war eine große Bereicherung.
Der größte und persönlichste Dank gilt meiner Familie. Meinen Eltern, Barbara und Robert Wrazidlo, danke ich von ganzem Herzen für ihre bedingungslose Unterstützung, nicht nur während dieser Abschlussphase, sondern während meines gesamten Studiums. Ihr unerschütterlicher Glaube an mich und ihre Vorbildfunktion haben den Grundstein für alles gelegt. Meinem Bruder, Peter Wrazidlo, danke ich für die Ermutigung, stets dem nachzugehen, wofür ich wirklich brenne. Dass ich diese Arbeit heute abgeben kann, ist letztlich der Verdienst von euch allen.
Zuletzt danke ich meinen Freunden, die mich zu der Person gemacht haben, die ich heute bin. Ohne das Umfeld und die Unterstützung dieser Menschen wäre ich nicht dort, wo ich heute stehe.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

In diesem Kapitel wird der Rahmen der Bachelorarbeit beschrieben. Die Motivation, weshalb dieses Thema einen wichtigen Beitrag zur Wärmewende leisten kann, sowie die daraus resultierende Aufgabenstellung werden behandelt. Die aus der Aufgabenstellung resultierenden Herausforderungen bilden den Abschluss dieses Kapitels.

1.1 Motivation

Der Klimawandel stellt eine der größten Herausforderungen des 21. Jahrhunderts dar und ist wissenschaftlich umfassend belegt. Studien des Intergovernmental Panel on Climate Change (IPCC) zeigen eindeutig, dass menschliche Aktivitäten, insbesondere die Nutzung fossiler Brennstoffe, die Hauptursache für den globalen Temperaturanstieg sind [1]. Um die negativen Folgen des Klimawandels zu begrenzen, hat die Europäische Union bereits im Jahr 2008 mit dem Klima- und Energiepaket erstmals verbindliche Reduktionsziele für Treibhausgasemissionen festgelegt. Mit dem Europäischen Green Deal von 2019 wurde das Ziel der Klimaneutralität bis 2050 beschlossen. Dieses Ziel betrifft zentrale Wirtschaftssektoren wie Energie, Industrie, Verkehr, Gebäude, Landwirtschaft und Abfallwirtschaft. In der öffentlichen Debatte über die Energiewende liegt der Schwerpunkt häufig auf der Dekarbonisierung des Stromsektors, insbesondere auf dem Ausbau erneuerbarer Energien wie Wind- und Solarenergie. Ein ebenso bedeutender Bereich wird jedoch oft vernachlässigt: die Wärmewende. Aktuelle Daten zeigen, dass etwa 26.2 % der Energie Europas für Wohngebäude benötigt wird[2]. Zudem werden 50 % des gesamten Endenergieverbrauchs von Wohnhäusern in Europa für die Wärmeerzeugung aufgewendet, wobei ein erheblicher Anteil weiterhin aus fossilen Energieträgern wie Erdgas, Öl oder Kohle stammt [3]. Die Transformation des Wärmesektors stellt somit einen wesentlichen Baustein zur Erreichung der Europäischen Klimaziele dar. Aus diesem Grund ist in Deutschland das Wärmeplanungsgesetz in Kraft getreten. Dieses sieht vor, dass der Anteil klimafreundlicher Wärme in Wärmenetzen bis zum Jahr 2030 im bundesweiten Durchschnitt auf mindestens 50 % ansteigen soll.[4]

Parallel zu dieser globalen Herausforderung eröffnet sich in der ingenieurwissenschaftlichen Praxis ein wachsendes Potenzial durch die Nutzung bereits vorhandener Datenbestände. Im Zuge der Projektdokumentation und archivierung sichern Ingenieurbüros eine Vielzahl projektspezifischer Daten. Hierzu zählen beispielsweise Dimensionierungsunterlagen, Inbetriebnahmeprotokolle, Herstellerdaten technischer Komponenten sowie detaillierte Lastprofile und Energieverbrauchsanalysen. Diese Daten werden bislang häufig nur archiviert, obwohl sie erheblichen Mehrwert für zukünftige Planungen und Prognosen bieten könnten. Der Einsatz datenbasierter Methoden, insbesondere von Modellen des maschinellen Lernens (ML), ermöglicht es, Muster in historischen Projektdaten zu erkennen und daraus verbesserte fundierte Entscheidungen für neue Projekte zu erreichen. Dies kann zu einer höheren Planungssicherheit, genaueren Prognosen und effizienteren Projektabläufen führen.

An dieser Schnittstelle zwischen Klimaschutz und datengetriebener Ingenieurpraxis setzt das Unternehmen Theta Concepts mit seiner Arbeit an. Es entwickelt kommunale Wärmepläne, die eine langfristige Umstellung auf erneuerbare Wärmequellen ermöglichen sollen, etwa durch die Integration von Biogasanlagen, Geothermie, Luftwärmepumpen oder Fernwärmenetzen. Eine zentrale Herausforderung dieser Planung besteht darin, den aktuellen Nutzwärmebedarf einer Kommune/Stadt möglichst präzise zu ermitteln. Diese Bedarfsprognose bildet die Grundlage für die Dimensionierung zukünftiger Wärmeversorgungssysteme. Eine genauere Vorhersage des Wärmebedarfs erhöht nicht nur die Planungssicherheit, sondern trägt auch zur Wirtschaftlichkeit bei. Insbesondere können kostenintensive Fehlauslegungen vermieden werden: Eine Überdimensionierung führt zu vermeidbaren Investitions- und Betriebskosten, während eine Unterdimensionierung aufwendige und teure Nachrüstungen erforderlich machen kann.

Derzeit stützt sich Theta Concepts bei der Bedarfsprognose auf das extern zugekaufte Programm Thermos. Durch den Abschluss mehrerer Projekte liegen dem Unternehmen inzwischen jedoch eigene Verbrauchsdaten und Gebäudeinformationen vor. Diese Daten bieten die Chance, ein internes Prognosewerkzeug auf Basis von ML-Modellen zu entwickeln. Damit könnte Theta Concepts unabhängiger von externen Anbietern werden, Planungskosten senken und potentiell die Qualität der Wärmepläne weiter verbessern.

Die vorliegende Bachelorarbeit verfolgt das Ziel, diese Möglichkeit exemplarisch zu untersuchen und dabei den Nutzen von ML in der Ingenieurpraxis aufzuzeigen. Sie verbindet damit zwei zentrale Anliegen: die Unterstützung der Wärmewende als Beitrag zum Klimaschutz sowie die bessere Nutzung vorhandener Projektdaten in Ingenieurbüros zur Steigerung von Effizienz und Prognosegenauigkeit.

Zudem verfolgt diese Arbeit das Ziel, als praxisorientierter Einstieg in das Thema ML zu dienen. Sie richtet sich an Leserinnen und Leser mit grundlegenden mathematischen Kenntnissen, die bisher keine tiefergehenden Erfahrungen im Bereich ML gesammelt haben.

1.2 Aufgabenstellung

Diese Arbeit untersucht, ob auf Grundlage der verfügbaren realen Verbrauchsdaten und Gebäudedaten ein verbessertes Prognosetool zur Schätzung des Nutzwärmebedarfs entwickelt werden kann. Mithilfe von ML-Modellen soll überprüft werden, inwieweit datengetriebene Modelle auf der vorhandenen Datengrundlage eine genauere Vorhersage liefern als das derzeit genutzte Programm Thermos.

Für die Entwicklung eines praxistauglichen Prognosemodells sind mehrere methodische Schritte erforderlich:

- **Datenaufbereitung**: Die erhobenen Verbrauchsdaten müssen strukturiert und für das Training eines ML-Modells aufbereitet werden.
- **Modellwahl**: Die Auswahl eines geeigneten ML-Ansatzes ist entscheidend, um die bestmögliche Prognosequalität zu erreichen.
- **Modelltraining und Validierung**: Das trainierte Modell muss umfassend getestet werden, um sicherzustellen, dass es für seinen spezifischen Anwendungsbereich verlässliche Ergebnisse liefert.

Diese Aspekte werden in der Arbeit detailliert betrachtet, um eine fundierte Bewertung der Einsatzmöglichkeiten von ML in der Wärmebedarfsprognose zu ermöglichen. Zudem wird der Forschungsstand zur Prognose des Nutzwärmebedarfs von Gebäuden mittels ML-Modellen thematisiert. Auf dieser Grundlage werden die Schritte der Datenaufbereitung, die Auswahl und das Training der ML-Modelle sowie deren anschließende Evaluierung erläutert. Anhand der Ergebnisse wird abschließend beurteilt, ob ein ML-Modell entwickelt wurde, das Thermos ersetzen kann.

2 Grundlagen der ML-Modelle

Um die Thematik der ML-Modelle in einer nachvollziehbaren Struktur zu erläutern, wird zunächst die Definition von ML-Modellen behandelt. Darauf aufbauend folgt die Beschreibung der allgemeinen Vorgehensweise beim Training von Modellen. Im Anschluss wird erläutert, wie ML-Modelle evaluiert werden und warum dieser Schritt von zentraler Bedeutung ist. Abschließend werden potenzielle Einflussfaktoren und Verzerrungen (Biases) im Trainingsprozess und bei der Datenakquise thematisiert. Diese Aspekte vervollständigen das Verständnis der grundlegenden Konzepte von ML-Modellen.

2.1 Definition von ML-Modellen

Ein Algorithmus, der in Datensätzen Muster erkennen soll, wird als ML-Modell bezeichnet. Dabei lassen sich diese Modelle in drei Hauptkategorien unterteilen.

Eine dieser drei Kategorien ist das überwachte Lernen (Supervised Learning). Hier werden dem ML-Modell feste Eingangs- und Ausgabegrößen vorgegeben. Das ML-Modell sucht dann, mit einer vom jeweiligen Modell abhängigen Methode, nach einem Muster, um den Zusammenhang zwischen diesen beiden Größen zu beschreiben [5, S. 23].

Beim bestärkenden Lernen (Reinforcement Learning) erhält das ML-Modell Eingaben in Form des aktuellen Zustands der Umgebung. Dem ML-Modell werden Aktionsmöglichkeiten gegeben mit dem Ziel, eine Aufgabe zu erfüllen. Es wählt daraufhin Aktionen und lernt durch Belohnung oder Bestrafung, welche Aktionen zum Erfolg führen. Ein gutes Beispiel hierfür ist das Finden des Weges durch ein Labyrinth. Das ML-Modell erhält anfangs die Information, wo es sich befindet und welche Aktionsmöglichkeiten es hat. Sobald das ML-Modell einen Weg gefunden hat, gibt es diesen aus [5, S. 23].

Liegen dem ML-Modell zwar Eingabegrößen, jedoch keine festen Ausgabegrößen vor, spricht man von unüberwachtem Lernen (Unsupervised Learning). Diese Methode ist nützlich, wenn Daten in Kategorien eingeordnet oder Zusammenhänge vereinfacht werden sollen. Ein treffendes Beispiel ist die Analyse von MRT-Scans zur Erkennung von Anomalien wie z. B. Tumoren. Hierbei erkennt das Modell Muster in den Bilddaten, die

auf potenzielle Auffälligkeiten hinweisen, ohne dass diese vorher explizit als Tumor gekennzeichnet wurden [5, S. 23].

In dieser Projektarbeit liegt der Fokus auf dem Supervised Learning. Diese Art von ML-Modellen eignet sich besonders für Optimierungsprozesse und wird verwendet, wenn zusammenhängende Eingabe- und Ausgabegrößen bereits vorliegen, was häufig bei archivierten Ingenieurdaten der Fall ist [6, S. 63].

Das passende ML-Modell für ein spezifisches Problem wird aus dem Hypothesenraum ausgewählt. Der Hypothesenraum umfasst alle mathematischen Modelle, die den Zusammenhang zwischen den Eingabe- und den Ausgabegrößen beschreiben könnten. Der Hypothesenraum lässt sich in zwei Hauptkategorien unterteilen:

1. **Lineare Hypothesen**: Diese Modelle beschreiben Zusammenhänge, die durch lineare Funktionen dargestellt werden können.

2. **Nicht-lineare Hypothesen**: Hierbei handelt es sich um Modelle, die komplexe, nicht-lineare Beziehungen zwischen den Variablen erfassen können.

Darüber hinaus werden ML-Modelle in zwei weitere Kategorien unterteilt, basierend auf der Art der Vorhersageaufgabe:

1. **Klassifikationsmodelle**: Diese Modelle ordnen Datenpunkten spezifische Kategorien oder Klassen zu. Ein Beispiel wäre die Klassifizierung von E-Mails als „Spam“ oder „Nicht-Spam“[7, S. 29].

2. **Regressionsmodelle**: Im Gegensatz dazu liefern Regressionsmodelle kontinuierliche numerische Werte, wie etwa die Vorhersage von Temperaturen oder Preisen [7, S. 29].

Sobald das ML-Modell ausgewählt ist, kann der Ablauf des Trainingsprozesses definiert werden. Das folgende Kapitel befasst sich mit der allgemeinen Vorgehensweise des Trainings und seinen Besonderheiten.

2.2 Trainingsprozess

Bevor das Training des ML-Modells beginnt, muss der zugrunde liegende Datensatz in Trainings- und Testdaten unterteilt werden. Alle Daten des Datensatzes, die zum Trainieren und Testen verwendet werden, enthalten die vollständigen Eingangs- und Ausgabegrößen. Typischerweise erfolgt diese Aufteilung im Verhältnis 80:20, wobei 80 % der Daten für

das Training und 20 % für die Evaluierung des Modells verwendet werden. Ein höherer Anteil an Trainingsdaten trägt dazu bei, dass das Modell die zugrunde liegenden Muster und Strukturen der Daten besser erlernen kann. Ein größerer Anteil an Testdaten wiederum ermöglicht eine differenziertere Analyse der Modellleistung. Da ein unzureichender Trainingsanteil typischerweise zu ungenauen oder unbrauchbaren Modellergebnissen führt, ist es essenziell, den Fokus auf eine ausreichend große Trainingsdatenmenge zu legen, um die Leistungsfähigkeit und Aussagekraft des Modells zu erhöhen [7].

Ein Datensatz besteht aus einer Sammlung von Datenpunkten, wobei jeder einzelne eine Reihe von Eingabegrößen und eine zugehörige Ausgabegröße (die Zielgröße) enthält. Das Ziel des ML-Modells ist es, zu lernen, wie die Eingabegrößen gemeinsam die Ausgabegröße beeinflussen. Diese zur Vorhersage verwendeten Eingabegrößen werden als Features bezeichnet. Beispielsweise können bei der Vorhersage des Verkaufspreises eines Autos das Baujahr, die Marke oder die Motorleistung als Features dienen, da diese Eigenschaften typischerweise Einfluss auf den Preis haben. Während des Trainingsprozesses wird jedem Feature interne Modellparameter zugewiesen. Diese Parameter beschreibt die Stärke und Richtung des Einflusses des jeweiligen Merkmals auf die Zielgröße und wird so angepasst, dass die Vorhersageleistung des Modells optimiert wird.

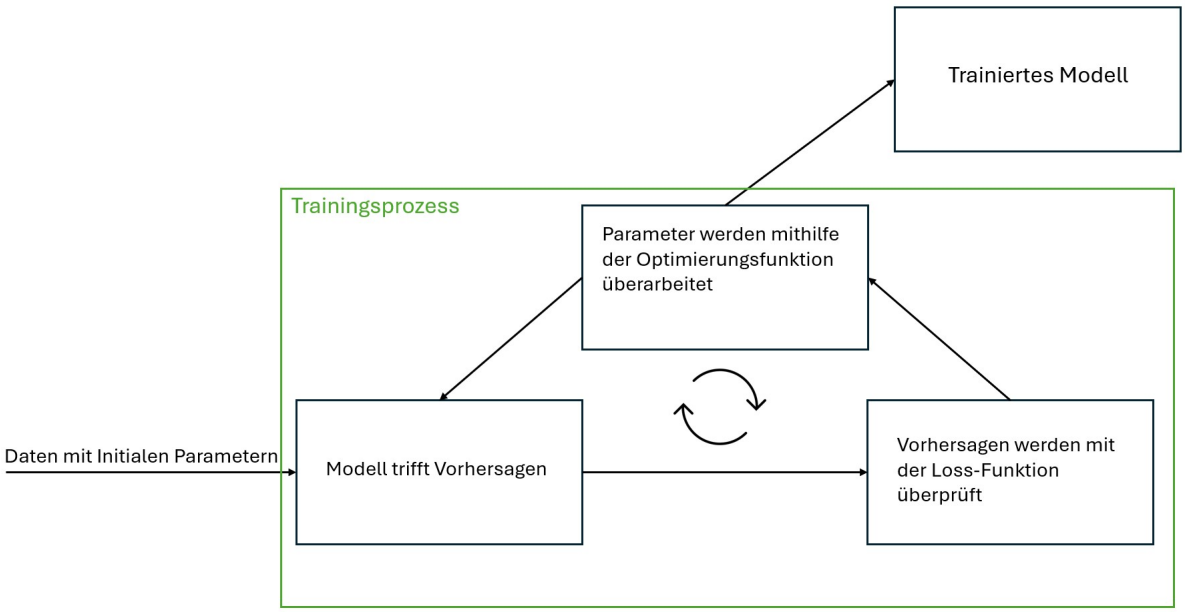

Abbildung 2.2.1: Verallgemeinerte Visualisierung des gesamten Trainingsprozesses eines ML-Modells.

Zu Beginn des Trainingsprozesses (siehe Abbildung 2.2.1) werden den Parametern des

Modells zunächst zufällige Startwerte zugewiesen. Basierend auf den Features und deren initialen Parametern trifft das Modell eine erste Vorhersage der Ausgabegröße. Die vorhergesagten Ausgabegrößen werden anschließend mithilfe der Loss-Funktion bewertet. Eine Loss-Funktion ist eine mathematische Formel, welche die Abweichung der vorhergesagten Ausgabegrößen von den in den Trainingsdaten hinterlegten realen Ausgabegrößen quantifiziert. Das Ziel des Trainingsprozesses ist es, den Wert dieser Funktion zu minimieren. Dies geschieht im dritten Schritt des Trainingsprozesses durch das ausführen der Optimierungsfunktion. Die Optimierungsfunktion passt die Parameter der Features so an, dass der Wert der Loss-Funktion sinkt.

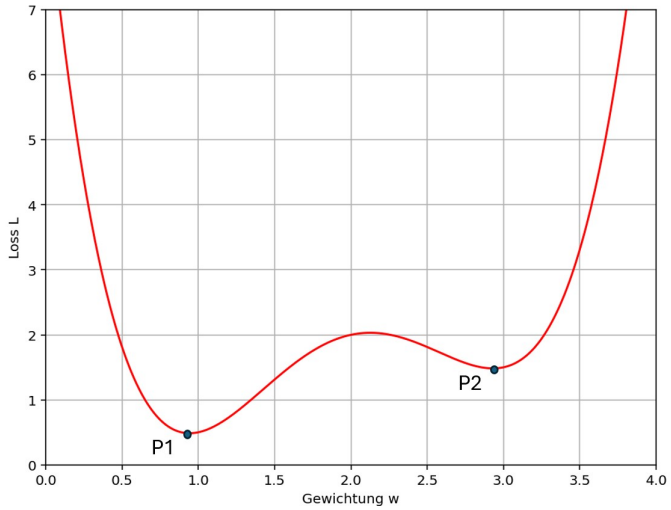

Abbildung 2.2.2: Beispielhafte Visualisierung einer Loss-Funktion einer Linearen Regression in Abhängigkeit von einem einzelnen Feature und damit einer einzelnen Gewichtung w. P1 ist das globale Minimum und P2 ein lokales Minimum.

Im mathematischen Kontext der Minimierung der Loss-Funktion besteht das Ziel darin, durch die Variation der Internen Parameter das globale Minimum der Funktion zu finden (P1 in Abbildung 2.2.2). In Abbildung 2.2.2 ist zur vereinfachten Erklärung eine Loss-Funktion einer Linearen Regression (LinReg) aufgetragen. Das Beispiel wurde ,da Bei der Linearen Regression die Gewichtung w der einzige Interne Parameter ist. In diesem Beispiel ist die Ausgangsgröße nur von einem Feature, also nur von dessen Gewichtung, abhängig ist. Je nach Loss-Funktion können neben dem globalen Minimum auch lokale Minima existieren (P2 in Abbildung 2.2.2). Beim Training ist es wichtig, eine Optimierungsfunktion zu wählen, die mit der gegebenen Loss-Funktion effizient umgehen kann. Andernfalls kann es passieren, dass das Modell in einem lokalen Minimum konvergiert,

anstatt das globale Minimum zu erreichen [7].

Der Trainingsdatensatz wird nicht als Ganzes in den Trainingsprozess eingeführt, sondern in kleineren Portionen, den sogenannten Batches. Dies hat den Vorteil, dass die Internen Parameter häufiger aktualisiert werden, wodurch der Trainingsprozess stabilisiert wird [8, S. 12]. Nachdem der gesamte Datensatz einmal komplett durch den Trainingsprozess geführt wurde, spricht man von einer Epoche. In der Regel werden Modelle über mehrere Epochen trainiert, um die Loss-Funktion so weit wie möglich zu minimieren [7, S. 113].

Das trainierte Modell wird im Anschluss mit dem Testdatensatz evaluiert. Dabei können verschiedene Bewertungsmaßstäbe angelegt werden.

2.3 Testprozess

Die Evaluierung eines ML-Modells hängt stark vom Anwendungsfall ab. Je nachdem, was der Zweck des ML-Modells ist, stehen verschiedene Aspekte im Vordergrund.

Bei der Beurteilung von Forschungsergebnissen, die ML-Modelle verwenden, um Zusammenhänge nachzuweisen, stehen statistische Aussagen im Vordergrund. Dort wird unter anderem auf die Signifikanz der einzelnen Features geachtet. In der Regel gilt ein Feature als signifikant, wenn sein p-Wert unter 0,05 liegt. Zudem wird auf die Modellinterpretierbarkeit geachtet. Das bedeutet, es ist wichtig nachvollziehen zu können, wie das Modell zu seinem Ergebnis gelangt. Ein weiterer wichtiger Wert in diesem Fachgebiet ist der R^2-Wert. Dieser gibt an, welcher Anteil der Varianz der Ausgabegröße durch das Modell erklärt werden [9].

In der praktischen Anwendung wird zusätzlich darauf geachtet, dass das trainierte Modell generalisierbar ist. Ein Modell besitzt eine gute Generalisierbarkeit, wenn es auf neue, ungesehene Daten angewendet werden kann und dabei die Genauigkeit beibehält, die es mit den Testdaten erreicht hat. Mit Genauigkeit sind in diesem Fall vor allem zwei Bewertungskriterien gemeint. Eines ist der durchschnittliche prozentuale Fehler zwischen der Vorhersage des Modells und der im Datensatz hinterlegten Ausgabegröße. Die anderen sind der Mean Square Error (MSE) und der Root Mean Square Error (RMSE). Diese beiden Metriken beziehen sich auf den absoluten Fehler und quantifizieren den durchschnittlichen quadratischen Unterschied zwischen der Vorhersage und der realen Ausgabegröße. Aufgrund ihrer Definition sind diese beiden Werte auf Regressionsmodelle beschränkt [10, S. 238].

Da sich diese Arbeit auf die Nutzung von ML-Modellen im Ingenieurbereich bezieht,

werden die für diese Anwendung relevanten Bewertungsmaßstäbe thematisiert.

Um die Generalisierbarkeit des ML-Modells anhand des gegebenen Datensatzes noch präziser zu bewerten, wird zusätzlich die K-Fold-Cross-Validation (KFC-Validation) durchgeführt.

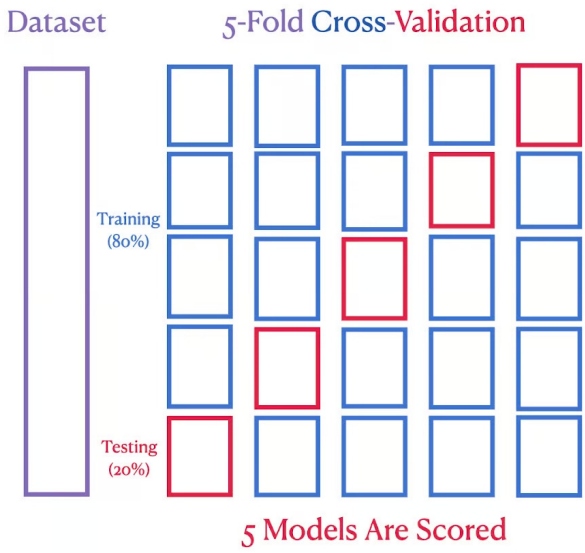

Abbildung 2.3.1: Visualisierung der KFC-Validation-Methode. Jeder Teil (in diesem Fall 20 % des Datensatzes) wird dabei einmal für Testzwecke benutzt.

Bei der KFC-Validation (Abbildung 2.3.1) wird der Datensatz in k gleich große Teilmengen (sogenannte *Folds*) aufgeteilt. Das Modell wird anschließend k-mal trainiert. In jedem Durchgang werden k-1 Folds als Trainingsdaten und der verbleibende Fold als Testdatensatz verwendet. Dieser Prozess wird so oft wiederholt, bis jeder Fold genau einmal als Testdatensatz gedient hat. Nach jedem Durchlauf wird eine Leistungskennzahl berechnet, typischerweise der MSE. Am Ende der KFC-Validation wird der durchschnittliche MSE über alle Folds ermittelt. Weicht dieser gemittelte MSE nicht signifikant (z. B. nicht mehr als 15 %) von dem MSE des ursprünglichen Testprozesses ab, kann das Modell als gut generalisierbar angesehen werden.

Ein wesentlicher Aspekt der KFC-Validation ist die zufällige Durchmischung des Datensatzes vor der Aufteilung in Folds. Diese Durchmischung verhindert, dass eine systematische Reihenfolge im Datensatz die Ergebnisse verfälscht. Eine Ausnahme bildet der Fall von unausgeglichenen Datensätzen. Datensätze sind unausgeglichen, wenn bestimmte Datenklassen überrepräsentiert sind. Ein Beispiel hierfür wäre ein Datensatz, der zu 90 % aus Daten von Frauen besteht, aber zum Vergleich der täglichen Schritte von Männern

und Frauen dienen soll. Wenn die Klassenverteilung im Datensatz stark unausgewogen ist, kann eine zufällige Aufteilung dazu führen, dass unterrepräsentierte Gruppen in einzelnen Folds zu stark oder zu wenig vertreten sind, was zu einer verzerrten Bewertung der Generalisierbarkeit führen würde. In solchen Fällen wird die *Stratified KFC-Validation* angewendet. Hierbei wird sichergestellt, dass die Klassenverteilung in jedem Fold proportional zur Verteilung im Gesamtdatensatz ist. Dies ermöglicht eine repräsentativere Bewertung der Generalisierbarkeit [11].

Um das Bild eines kompletten Trainings- und Testprozesses abzurunden, werden nun Biases thematisiert. Hierbei wird erklärt, welche Biases durch die Zusammenstellung des Datensatzes entstehen können und welche als Bias interpretierbaren Verhaltensmuster sich in trainierten ML-Modellen erkennen lassen.

2.4 Biases

Um systematisch vorzugehen, werden in diesem Kapitel zuerst die Biases auf Datenebene thematisiert, um danach die trainingsbasierten Biases zu erläutern.

Der Datensatz ist das Fundament des ML-Modells. Die Qualität dieser Daten entscheidet maßgeblich darüber, wie sich das Modell bei neuen Daten verhalten wird. Der Grund dafür ist, dass ein generalisierbares ML-Modell für die gesamte Bandbreite an Daten, für die es später Vorhersagen treffen soll, eine ausreichende Menge an Trainingsdaten benötigt. Ebenfalls müssen beim Testen des Modells für die gewählte Bandbreite Daten zur Verfügung stehen, mit denen die Genauigkeit des Modells für den jeweiligen Bereich beurteilt werden kann.

Wenn der Datensatz diese Eigenschaft nicht in ausreichendem Maße besitzt, liegt ein Stichproben-Bias (Sampling Bias) vor. Ein Beispiel hierfür wäre ein ML-Modell, das für die Gesichtserkennung eingesetzt werden soll. Wenn das ML-Modell nur mit Gesichtern sehr alter Personen trainiert wurde, wird es jüngere Gesichter schwerer oder gar nicht erkennen. Deshalb ist bei der Datenauswahl darauf zu achten, dass die im Fokus liegenden Kategorien auch im Datensatz repräsentiert sind [12].

Wenn bei der Datenerhebung durchgängig derselbe Messfehler auftritt, spricht man von einem systematischen Bias. Dies bedeutet, dass alle Messwerte in eine bestimmte Richtung verschoben sind (siehe Abbildung 2.4.1, rechts). Auf der Zielscheibe im Bild sind die Einschläge zwar eng beieinander (geringe Streuung), aber nicht im Zentrum. Ein Beispiel wäre ein Messinstrument, das konstant zu hohe oder zu niedrige Werte liefert. Dadurch entsteht eine Verzerrung, die nicht durch mehr Messungen ausgeglichen werden

Abbildung 2.4.1: Beispiel für zufälligen Bias (mittig), systematischen Bias (rechts) und keinen Bias (links) [13]

kann. Beim zufälligen Bias hingegen (Abbildung 2.4.1, mittig) sind die Einschläge zwar um das Zentrum verteilt, aber nicht eng beieinander. Das entspricht einem Messverfahren, das eine Ungenauigkeit aufweist, sodass die Werte in einem gewissen Bereich streuen. Systematische Biases stellen ein größeres Problem dar als zufällige Biases. Während zufällige Fehler durch die Mittlung mehrerer Messungen reduziert werden können, bleibt ein systematischer Fehler bestehen. Da die Messwerte eine kleine Varianz aufweisen, kann fälschlicherweise angenommen werden, dass sie präzise sind, obwohl sie nicht korrekt sind [12].

Nun werden die Biases thematisiert, die direkt mit dem ML-Modell zusammenhängen: Unter- und Überanpassung.

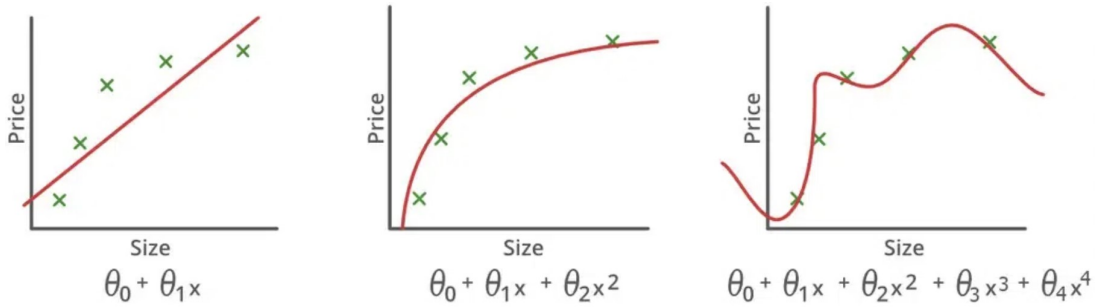

Abbildung 2.4.2: Beispiel für Unteranpassung (links), Überanpassung (rechts) und eine gute Kombination aus Daten und ML-Modell (mittig) [14]

Die Unteranpassung (Underfitting, siehe Abbildung 2.4.2, links) beschreibt ein ML-Modell, das einen zu simplen mathematischen Zusammenhang zwischen Eingabe- und Ausgabegrößen eines Datensatzes abbildet. Dabei ignoriert das Modell einen großen Teil

der Komplexität des Datensatzes und trifft ungenaue Vorhersagen. Ein solches Verhalten deutet auf ein Modell hin, das nicht komplex oder sensitiv genug ist.

Bei der Überanpassung (Overfitting, siehe Abbildung 2.4.2, rechts) wurde ein ML-Modell gewählt, das zu komplex ist, um die zugrunde liegenden Zusammenhänge in den Daten zu beschreiben. Durch die zu hohe Komplexität reagiert das ML-Modell stark auf zufällige Schwankungen, da es vermeintliche Muster im Datensatz erkennt. Diese Schwankungen sind in Datensätzen immer vertreten und werden als Rauschen bezeichnet. Rauschen beschreibt die Streuung der Daten, die nicht durch die Eingabegrößen erklärt werden kann. Ein Beispiel hierfür wäre die Messung der Körpergröße einer Person: Wenn über mehrere Tage die Größe derselben Person erfasst wird, können die Messwerte abhängig von der Körperhaltung oder der Messuhrzeit schwanken. Da diese beiden Größen nicht erfasst wurden, kann der Datensatz die Unterschiede der Messungen nicht erklären; sie werden daher als Rauschen im Datensatz wahrgenommen. Während der Trainingsphase des Modells ist die Überanpassung oft nicht direkt erkennbar. Wenn das Modell jedoch auf neue Daten angewendet wird, sinkt seine Genauigkeit drastisch. Das passiert, da es sich an die spezifischen Eigenheiten des Trainingsdatensatzes inklusive seines Rauschens überangepasst hat und mit diesen gelernten Mustern auf einem neuen Datensatz mit dessen eigenem Rauschen nicht zurechtkommt [12]. Bei der Wahl eines Modells, das ein gutes Maß an Komplexität aufweist (Abbildung 2.4.2, mittig), können später genaue Vorhersagen für neue Daten getroffen werden.

Nachdem nun die relevanten Grundlagen von Trainings- und Testprozessen für ML-Modelle dargelegt wurden, kann auf Projekte eingegangen werden, die sich bereits mit der Prognose des Nutzwärmebedarfs von Gebäuden beschäftigen.

3 Stand der Technik

Es wurden bereits viele verschiedene Herangehensweisen verwendet, um mittels ML-Modellen eine Aussage über den Nutzwärmebedarf eines Gebäudes zu treffen.

In einer Studie aus dem Jahr 2015 wurde untersucht, wie gut sich der Heizbedarf von Universitätsgebäuden mithilfe verschiedener neuronaler Netzwerke vorhersagen lässt. Dabei kamen ein Feedforward-Backpropagation-Netzwerk (FFNN), ein Radial-Basis-Function-Netzwerk (RBFN) und ein Adaptive-Neuro-Fuzzy-Inference-System (ANFIS) zum Einsatz. Als Eingangsgrößen wurden ausschließlich meteorologische Daten genutzt, darunter die mittlere tägliche Außentemperatur, die maximale und minimale Temperatur, die Windgeschwindigkeit, die relative Luftfeuchtigkeit, die tägliche solare Einstrahlung, der Wochentag sowie der Heizenergieverbrauch des Vortages. Die Datenbasis umfasste insgesamt etwa 400 Stichproben, von denen rund 300 für das Training und 100 für das Testen der Modelle verwendet wurden. Die Modelle erzielten eine mittlere prozentuale Abweichung zwischen 3,4 % und 6 %, wobei die Ensemble-Methoden durch die Kombination mehrerer Netzwerke die besten Ergebnisse lieferten. Die Studie zeigt, dass für Gebäude mit ähnlichen geometrischen Eigenschaften präzise Vorhersagen des Heizenergieverbrauchs möglich sind. Allerdings sind die Modelle nicht in der Lage, den Verbrauch einzelner Gebäude mit stark unterschiedlichen geometrischen Merkmalen zuverlässig vorherzusagen, da diese zusätzlichen Einflussfaktoren nicht berücksichtigt wurden [15].

Eine weitere Studie aus dem Jahr 2017 beschäftigte sich mit dieser Thematik. Dabei wurden robustere ML-Modelle wie die LinReg eingesetzt, um ein generalisierbares Modell zu entwickeln. Als Eingangsgrößen wurden sowohl geometrische als auch umweltbezogene und gebäudespezifische Faktoren berücksichtigt. Dazu gehörten unter anderem der Wärmedurchgangskoeffizient eines Gebäudes, das Verhältnis von Fenster- zu Bodenfläche, der Gebäudeformfaktor sowie die durchschnittliche Temperaturdifferenz zwischen Innen- und Außenraum. Die Untersuchung ergab, dass die Vorhersagen um bis zu 20 % von den realen Verbrauchswerten abwichen. Die Studie betonte, dass für ML-Modelle, die eine hohe Generalisierbarkeit in diesem Anwendungsbereich aufweisen sollen, eher komplexere Methoden wie die Support Vector Regression (SVR) verwendet werden sollten [16].

Ein Projekt, das diesen Ansatz verfolgte, war das Thermos-Projekt. Das Thermos-Projekt (Thermal Energy Resource Modelling and Optimisation System) wurde im Rahmen der EU-Initiative „Horizon 2020“ entwickelt, um Kommunen und Energieversorgern eine effiziente Planungsplattform für die Optimierung von Fernwärme- und Fernkältenetzen bereitzustellen. Das zugrundeliegende Modell zur Vorhersage des jährlichen Nutzwärmebedarfs wurde ausschließlich auf Basis von Daten aus Kopenhagen erstellt, da dort eine hohe Verfügbarkeit detaillierter Geodaten und realer Verbrauchsdaten aus dem bestehenden Fernwärmenetz gegeben war. Zur Modellierung wurden ausschließlich fünf geometrische Eingangsgrößen verwendet: der Umfang, die Höhe, das Volumen, die Grundfläche sowie die Außenwandfläche eines Gebäudes. Mithilfe von SVR und LinReg wurden diese Parameter genutzt, um den jährlichen Nutzwärmebedarf zu prognostizieren. Die Genauigkeit des Modells zeigte eine durchschnittliche Abweichung von 10–15 % gegenüber den realen Verbrauchswerten, womit es eine robuste Grundlage für die Wärmebedarfsabschätzung darstellt [17].

Aus diesen Forschungen ergibt sich der methodische Rahmen für die Auswahl geeigneter ML-Modelle. Da das Ziel dieser Arbeit die Entwicklung eines generalisierbaren ML-Modells ist, liegt der Fokus auf robusten Modellierungsansätzen wie der LinReg und der SVR. Aufgrund ihrer nachgewiesenen Stabilität und Leistungsfähigkeit in vergleichbaren Studien bieten diese Modelle eine geeignete Grundlage für die weitere Analyse. Im folgenden Kapitel werden daher die technischen Grundlagen dieser Methoden detailliert erläutert. Diese Konzepte bilden die Basis für das Verständnis der im Rahmen dieser Bachelorarbeit entwickelten Modelle und ermöglichen eine fundierte Nachvollziehbarkeit der getroffenen Modellierungsentscheidungen.

4 Verschiedene ML-Modelle

Aus den Ergebnissen des vorhergehenden Kapitels 3 geht hervor, dass sich insbesondere die LinReg und die SVR als geeignete Modelle für den Anwendungsbereich dieser Bachelorarbeit herausgestellt haben. Beide Regressionsverfahren zählen zu den überwachten Lernverfahren und finden in der industriellen Praxis breite Anwendung [18].

Für jedes der betrachteten ML-Modelle erfolgt eine detaillierte mathematische Beschreibung der Modellstruktur, einschließlich der zugrunde liegenden Annahmen. Darüber hinaus werden die spezifischen Eigenschaften jedes Modells herausgearbeitet, wobei sowohl deren Vorteile als auch Nachteile diskutiert werden. Diese umfassende Betrachtung ermöglicht eine fundierte Bewertung der beiden Regressionsmethoden und bietet eine ganzheitliche Zusammenfassung ihrer Stärken und Schwächen.

4.1 Lineare Regression

Die Grundlagen für dieses ML-Modell wurden von Adrien-Marie Legendre im Jahr 1805 gelegt [19]. Bei diesem ML-Modell wird die generelle Annahme getroffen, dass die Ausgabegröße als lineare Kombination der Eingabegrößen dargestellt werden kann. Die Funktionsweise des Modells lässt sich am besten verstehen, indem man zunächst die Berechnung für eine einzelne Vorhersage betrachtet. Die mathematische Formulierung in dieser skalaren Schreibweise lautet:

$$\hat{y}_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} \tag{4.1}$$

Hierbei ist $\hat{y}_i$ der vom Modell vorhergesagte Wert für die i-te Beobachtung. Die Werte $x_{i1}, \ldots, x_{ip}$ sind die zugehörigen Ausprägungen der p Features. Der Parameter β_0 repräsentiert den y-Achsenabschnitt (Bias), während $\beta_1, \ldots, \beta_p$ die Gewichtungsfaktoren. sind Für die weitere Erläuterung dieses ML-Modells wird die Vektorschreibweise verwendet.

Hierzu werden alle Ausgabegrößen der einzelnen$\hat{y}_i$ zu einem Vektor $\hat{y}$ zusammengefasst:

$$\hat{y} = \mathbf{X}\boldsymbol{\beta} \tag{4.2}$$

Hierbei werden die Eingabegrößen in einer Matrix $\mathbf{X}$ zusammengefasst, in der jede Zeile einen Datenpunkt und jede Spalte eine Eingabegröße darstellt. Die jeweiligen Gewichte $\boldsymbol{\beta}$ sind in einem einspaltigen Vektor zusammengefasst. Ausgeschrieben sieht diese Schreibweise folgendermaßen aus:

$$\hat{y} = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} \tag{4.3}$$

Wie bei der allgemeinen Definition eines ML-Modells wird beim Training versucht, die Differenz zwischen der vorhergesagten Ausgabegröße $\hat{y}$ und der realen Ausgabegröße y, auch als Loss-Funktion bezeichnet, zu minimieren:

$$\min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{4.4}$$

Die Differenz der beiden Größen wird quadriert, damit sich negative und positive Abweichungen nicht gegenseitig aufheben und größere Fehler stärker gewichtet werden. Eine alternative Darstellung ist die Matrix-Schreibweise, die auf der quadratische euklidischen Norm basiert:

$$\min \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \tag{4.5}$$

Mit diesen mathematischen Grundlagen des LinReg-Modells kann nun die Optimierung dieser Funktion beschrieben werden [7, S. 19].

Da es sich hierbei um eine konkave quadratische Funktion handelt, existiert nur ein globales Minimum und keine lokalen Minima. Aufgrund dieser Eigenschaft ist es möglich, das Optimum dieses ML-Modells durch Ableiten und Nullsetzen der Funktion zu finden. Dies erfordert deutlich weniger Rechenaufwand als der iterative Optimierungsprozess, der in Abbildung 2.2.1 dargestellt ist. Um das Minimum der Loss-Funktion zu finden, ist die einzige beeinflussbare Größe der Gewichtsvektor $\boldsymbol{\beta}$. Demnach wird nach dieser Größe abgeleitet, wodurch die Funktion $F'(\boldsymbol{\beta})$ entsteht. Wenn die euklidische Normschreibweise

ausgeschrieben wird, ergibt sich das zweite Binom:

$$F(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$
$$F(\boldsymbol{\beta}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \tag{4.6}$$

Wenn diese Funktion nach $\boldsymbol{\beta}$ abgeleitet wird, entsteht folgende Gleichung:

$$F'(\boldsymbol{\beta}) = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \tag{4.7}$$

Diese Funktion muss nun, um das globale Minimum zu finden, Null gesetzt und nach $\boldsymbol{\beta}$ aufgelöst werden. Dadurch ergibt sich folgende Formel:

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{4.8}$$

Diese Formel wird als Methode der kleinsten Quadrate (Ordinary Least Squares) bezeichnet und bestimmt eine Hyperebene, zu der die Summe der quadrierten Abstände aller Datenpunkte minimal ist [7]. Mit diesem neu bestimmten, optimalen Gewichtsvektor kann das LinReg-Modell auf neue Daten angewendet werden.

Tabelle 4.1: Vor- und Nachteile der Linearen Regression.

Vorteile	**Nachteile**
Hohe Interpretierbarkeit (Transparenz): Es ist nachvollziehbar, welchen Einfluss jede Eingangsgröße auf die Ausgabegröße hat.	Annahme eines linearen Zusammenhangs. Bei nicht-linearen Daten die nicht entsprechend vorbereitet wurden, kann dies zu Unteranpassung führen.
Geringe Berechnungszeit (schnelle Vorhersagen möglich) und keine Hyperparameter-Optimierung nötig.	Risiko von Verzerrungen, wenn zwischen den Eingangsgrößen eine starke Korrelation (Multikollinearität) besteht.
Eignet sich für die Vorhersage neuer Daten, wenn die Annahmen erfüllt sind.	Anfälligkeit gegenüber Ausreißern, die die Genauigkeit stark verzerren können.

Zur besseren Übersicht über die Vor- und Nachteile des Modells dient Tabelle 4.1. Falls jedoch kein linearer Zusammenhang zwischen den Eingangs- und Ausgabegrößen besteht, eignen sich flexiblere Modelle wie die SVR.

4.2 Support Vector Regression

Die SVR hat ihren Ursprung in den Support Vector Machines (SVM), die beide 1995 von Vladimir Vapnik und seinem Team entwickelt wurden. Während das ursprüngliche SVM-Modell primär für Klassifikationsaufgaben konzipiert war, wurde es zur Lösung von Regressionsproblemen durch das SVR-Modell erweitert [20].

Das Ziel des SVR-Modells ist es, wie bei der LinReg, eine Funktion zu entwickeln, um die Beziehung zwischen den Eingabe- und Ausgabegrößen zu beschreiben. Es gibt jedoch zwei wesentliche Unterschiede zum LinReg-Modell:

1. **Toleranzbereich**: Das SVR-Modell errichtet einen Toleranzbereich (auch als ϵ-Tube bezeichnet) um die Funktion herum. Fehler werden nur für Datenpunkte außerhalb dieses Bereichs bestraft.

2. **Kernel-Trick**: Das SVR-Modell kann mithilfe verschiedener Kernel-Funktionen auch nicht-lineare Zusammenhänge abbilden, indem die Daten in einen höherdimensionalen Raum transformiert werden.

Es existieren drei verbreitete Kernel-Funktionen für das SVR-Modell:

1. **Linearer Kernel**: Wird verwendet, wenn zwischen Eingabe- und Ausgabegrößen ein linearer Zusammenhang angenommen wird.

2. **Polynomieller Kernel**: Eignet sich für Zusammenhänge, die nicht streng linear, aber auch nicht hochkomplex sind.

3. **Radial Basis Function (RBF) Kernel**: Wird für komplexe, nicht-lineare Zusammenhänge verwendet. Er passt sich sehr flexibel an die Strukturen der Daten an.

Zur besseren Veranschaulichung des Toleranzbereichs dient Abbildung 4.2.1, die ein SVR-Modell mit linearem Kernel zeigt. Das Optimierungsprinzip ist für alle Kernel-Typen identisch. Ziel ist es, eine Funktion zu finden, bei der möglichst viele Datenpunkte innerhalb des Toleranzbereichs liegen. Die allgemeine Funktionsgleichung beim linearen Kernel ist identisch mit der des LinReg-Modells (Gleichung 4.2). Der Unterschied liegt in der Funktion, die für die Optimierung minimiert wird:

$$\min \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{4.9}$$

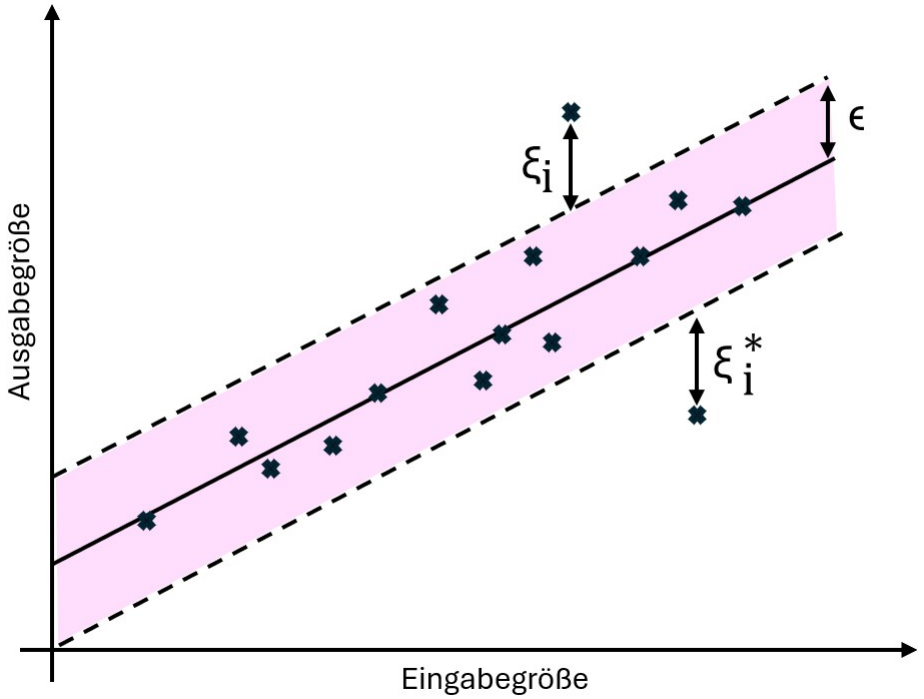

Abbildung 4.2.1: Visualisierung des Toleranzbereichs (pinker Bereich) eines SVR-Modells, festgelegt durch $\pm\epsilon$. Die Abstände der außerhalb liegenden Datenpunkte werden durch die Slack-Variablen ξ_i^* und ξ_i dargestellt.

Die Optimierung fokussiert sich auf zwei Aspekte. Zum einen wird die quadrierte Norm des Gewichtsvektors $\|\boldsymbol{\beta}\|^2$ minimiert. Dies dient als Regularisierung, um Überanpassung zu vermeiden. Der zweite Term minimiert die Summe der Abweichungen der Datenpunkte, die außerhalb des Toleranzbereichs ϵ liegen. Der Faktor C ist, ähnlich wie ϵ, ein benutzerdefinierbarer Hyperparameter. Er steuert wie Abweichungen außerhalb des Toleranzbereiches in die zu minimierende Funktion einfließt. Ein hoher Wert von C bestraft Abweichungen stark, was zu einer engeren Anpassung an die Trainingsdaten führen kann (Risiko der Überanpassung). Ein niedriger Wert von C erlaubt größere Abweichungen und führt zu einem robusteren Modell (Risiko der Unteranpassung). Die Rahmenbedingungen für die Optimierung werden durch folgende Nebenbedingungen definiert:

$$y_i - (\boldsymbol{\beta}^T \mathbf{x}_i + b) \leq \epsilon + \xi_i \tag{4.10}$$

$$(\boldsymbol{\beta}^T \mathbf{x}_i + b) - y_i \leq \epsilon + \xi_i^* \tag{4.11}$$

$$\xi_i, \xi_i^* \geq 0 \tag{4.12}$$

Hier wird mathematisch festgehalten: Wenn die Abweichung zwischen der Vorhersage ($\boldsymbol{\beta}^T\mathbf{x}_i + b$) und dem tatsächlichen Wert y_i kleiner oder gleich der Toleranz ϵ ist, sind die Slack-Variablen ξ_i und ξ_i^* null, und der Datenpunkt erzeugt keinen Fehler. Ist die Abweichung größer, wird ξ_i oder ξ_i^* positiv, und der Fehler fließt in die Optimierung ein. Bei diesem Modell ergibt sich anhand von Gleichung (4.9) ein quadratisches Optimierungsproblem mit linearen Nebenbedingungen. Da dieses Problem aufgrund seines konkaven Charakters keine lokalen Minima besitzt, ist ebenfalls keine iterative Optimierungsschleife (siehe Abbildung 2.2.1) erforderlich. Stattdessen wird die Optimierungsfunktion mithilfe der Lagrange-Methode gelöst [20].

Tabelle 4.2: Vor- und Nachteile der Support Vector Regression.

Vorteile	**Nachteile**
Effektiv bei hochdimensionalen Daten und wenn die Anzahl der Features größer ist als die der Datenpunkte.	Geringere Interpretierbarkeit (Black-Box-Charakter bei nicht-linearen Kerneln): Es ist nicht direkt nachvollziehbar, welchen Einfluss die einzelnen Eingangsgrößen haben.
Dank des Kernel-Tricks sehr flexibel bei der Modellierung nicht-linearer Zusammenhänge.	Die Wahl der Hyperparameter C und ϵ sowie des Kernels kann komplex sein und erfordert oft eine systematische Optimierung (z. B. Grid-Search).
Robust gegenüber Ausreißern, da Abweichungen nicht quadratisch eingehen wie z.B bei der LinReg	Potenziell lange Berechnungszeit bei großen Datensätzen.

5 Vorgehen

Im Folgenden wird das methodische Vorgehen dieser Arbeit in der chronologischen Reihenfolge dargestellt, wie es im Projektverlauf tatsächlich durchgeführt wurde. Beginnend mit der Datenakquise über die Aufbereitung und Auswahl relevanter Merkmale (Features) bis hin zur Evaluation verschiedener ML-Modelle wird der gesamte Analyseprozess nachvollziehbar beschrieben. Nach der Auswahl des am besten geeigneten Modells liegt der Fokus auf der Frage, wie sich dessen Generalisierbarkeit optimieren lässt, um auch auf unbekannten Daten zuverlässige Vorhersagen zu ermöglichen.

5.1 Datenakquise und aufbereitung

Die Datenakquise beginnt damit, dass die jeweilige Gemeinde Theta Concepts den Auftrag zur Erstellung eines Wärmeplans erteilt. Damit wird auch die Befugnis erteilt, auf vertrauliche Daten wie Einträge über die einzelnen Häuser aus dem Amtlichen Liegenschaftskataster-Informationssystem (ALKIS) zuzugreifen. Zusammen mit anderen frei zugänglichen Quellen über die Geometrie eines Hauses, wie OpenStreetMap, wird eine erste Datengrundlage geschaffen. Um die Höhe der Gebäude zu ermitteln, stellt die jeweilige Stadt Airborne-Laser-Scanning-Daten (ALS) zur Verfügung. Diese durch Flugzeuge aufgenommenen Laserdaten geben Auskunft über Gebäudehöhen. Die Daten der einzelnen Städte werden zur Weiterverarbeitung in einem einzigen Datensatz zusammengefasst.

Diese Datengrundlage erfordert anschließend einige Bearbeitungsschritte, um sie als Trainingsdatensatz für ein ML-Modell zu nutzen. Zunächst werden die Gebäude danach klassifiziert, ob sie einen potenziellen Wärmebedarf haben. Dies wird anhand von zwei Kriterien bestimmt. In den ALKIS-Daten wird über die Nutzung der einzelnen Gebäude Auskunft gegeben. Gebäude, die eindeutig nicht zu Wohnzwecken genutzt werden, können dadurch ausgeschlossen werden. Da diese Daten jedoch nicht immer vollständig sind, muss eine allgemeine Regel implementiert werden. Basierend auf den in mehreren Projekten gesammelten Erfahrungen hat sich eine Regel etabliert, welche Gebäude mit weniger als $30\,\mathrm{m}^2$ Grundfläche und einer Höhe von unter $3\,\mathrm{m}$ als Gebäude ohne Wärmebedarf einstuft.

Den übrigen Gebäuden muss nun ein Realverbrauch zugeordnet werden. Dabei werden Daten von Energieversorgern herangezogen, die im Rahmen der Projekte beschafft wurden. Diese Realverbräuche sind grundstücksscharf, wie in Abbildung 5.1.1 gezeigt. Dies liegt daran, dass die Energieversorger nicht danach differenzieren, welches Haus welchen Verbrauch hat, sondern nur, welche Rechnungsadresse welchen Gesamtverbrauch aufweist. Deshalb wird jedem Gebäude ein spezifischer Energiebedarf basierend auf seinem prozentualen Flächenanteil am Gesamtgrundstück zugeordnet.

Abbildung 5.1.1: Beispielhafte Darstellung der Problematik des grundstücksscharfen Realverbrauchs. Den vier Geometrien (als Kästchen mit schwarzer Umrandung) muss aus einem einzigen Gesamtenergieverbrauch ein individueller, gebäudescharfer Energieverbrauch zugeordnet werden.

Um ein generalisierbares Prognosemodell zu trainieren, muss der gemessene Realverbrauch eines Gebäudes zu einem klimabereinigten Nutzwärmebedarf normalisiert werden. Dieser Wert ist unabhängig von zwei verzerrenden Hauptfaktoren: den spezifischen Wetterbedingungen eines Jahres und der Effizienz der installierten Heizungsanlage. Die Klimabereinigung eliminiert den Einfluss von Witterungsschwankungen. Dazu wird der Realverbrauch mit einem Faktor korrigiert, der auf Heizgradtagen basiert. Die Heizgradtagzahl eines Jahres summiert die täglichen Temperaturdifferenzen unterhalb einer Heizgrenze (z. B. 15 °C) und misst so die „Kälte“ eines Jahres. Indem die durchschnittliche Heizgradtagzahl der letzten 20 Jahre durch die des Messjahres geteilt wird, erhält man einen Korrekturfaktor. Dieser normiert den Verbrauch, indem er ihn für ein überdurchschnittlich kaltes Jahr reduziert und für ein warmes Jahr erhöht. Im zweiten Schritt wird aus diesem klimabereinigten Energieverbrauch der Nutzwärmebedarf ermittelt. Hierbei wird der Wirkungsgrad der jeweiligen Heizungsanlage berücksichtigt, um zu bestimmen, wie viel Energie tatsächlich als Wärme im Gebäude ankommt. Der klimabereinigte Verbrauch wird dazu mit dem anlagenspezifischen Wirkungsgrad (z. B. 0,97 für Fernwärme oder 0,875 für Gas) multipliziert. Das Ergebnis ist ein standardisierter Bedarfswert, der einen fairen Vergleich zwischen Gebäuden ermöglicht und als robuste Zielgröße für das

ML-Modell dient. Häuser, die durch andere Heizarten versorgt werden, konnten aufgrund mangelnder Daten nicht berücksichtigt werden.

Da nun noch alle Gebäude mit Energiebedarf enthalten sind, müssen jene, die industriell, wirtschaftlich oder gemeinschaftlich (z. B. Schulen) genutzt werden, entfernt werden. Da das Modell auf die Prognose von Wohngebäuden abzielt, würden diese Gebäude die Daten verzerren. Ebenso können Gebäude mit sehr geringem Energieverbrauch, wie etwa nur zeitweise genutzte Ferienhäuser oder Gebäude mit einer nicht erfassten zweiten Wärmequelle, die Datenqualität beeinträchtigen. Deshalb wurden die Daten durch manuelle Untersuchung und die Anwendung einer Regel, keine Gebäude mit einem Bedarf unter 5 MWh zu inkludieren, bereinigt. Die Grenze von 5 MWh beruht auf Erfahrungswerten aus früheren Projekten.

Mit den nun aufbereiteten Daten kann die Suche nach dem bestgeeigneten Modell beginnen.

5.2 Modelltraining und evaluation

Der erste Trainingsansatz beider ML-Modelle ist eine Brute-Force-Methode. Dabei werden alle verfügbaren Features sowie gegebenenfalls verschiedene Auswahlmöglichkeiten für die Hyperparameter der ML-Modelle zur Verfügung gestellt. Folgende Features stehen den ML-Modellen zur Verfügung:

1. **Umfang**: Abgeleitet aus dem Umriss der Geometrie.
2. **Grundfläche**: Abgeleitet aus der Fläche der Geometrie.
3. **Zusammenhängender Umfang**: Abgeleitet aus dem Umfang, der von zwei Geometrien gemeinsam genutzt wird (z. B. bei Reihenhäusern).
4. **Höhe**: Erhalten durch ALS-Daten.
5. **Außenwandfläche**: Errechnet durch (Umfang − Zusammenhängender Umfang) · Höhe.
6. **Volumen**: Errechnet durch Grundfläche · Höhe.
7. **Nutzgrundfläche**: Errechnet nach DIN 277 durch die Formel: $0,32\,\mathrm{m}^{-1} \cdot$ Volumen. Sie soll ein Maß dafür sein, welche Fläche tatsächlich beheizt wird [21].

Das Modell wird mit einer Datenaufteilung von 80 % Trainings- und 20 % Testdaten mit jeder möglichen Kombination dieser Parameter trainiert und evaluiert. Die Kennzahl, an der die Leistung des Modells gemessen wird, ist hierbei der RMSE. Da der Energiebedarf über eine hohe Spannweite prognostiziert wird, könnten allein durch diese Prüfmethode große Gebäude mit hohem Energieverbrauch – und somit auch höheren absoluten Differenzen – das Ergebnis stark verzerren. Deshalb werden die Ergebnisse in vier Kategorien unterteilt:

1. **Gebäude von 0–100 MWh**: Hierzu zählen eher Einfamilienhäuser oder Doppelhaushälften.
2. **Gebäude von 100–500 MWh**: Damit werden Mehrfamilienhäuser oder kleinere Hochhäuser untersucht.
3. **Gebäude >500 MWh**: Besonders große, zusammenhängende Wohnblöcke.
4. **Gesamtdatensatz**: Um einen Gesamteindruck zu gewinnen.

Somit kann der Verzerrung durch die Betrachtung der einzelnen Kategorien entgegengewirkt werden. Die Varianten der ML-Modelle, die hinsichtlich der Ergebnisse am besten abschneiden, werden anhand von detaillierten Boxplot-Graphen untersucht und mit dem momentan genutzten Thermos-Modell verglichen.

Abbildung 5.2.1 visualisiert einen Leistungsvergleich zwischen dem selbst entwickelten ML-Modell (blaue Boxplots) und dem Referenzmodell von Thermos (rote Boxplots). Die x-Achse kategorisiert die Gebäude anhand ihres klimabereinigten Nutzwärmebedarfs in diskrete Intervalle von 10 MWh Breite. Die y-Achse stellt die prozentuale Abweichung der Modellvorhersage vom tatsächlichen Verbrauch dar, wobei der Nullpunkt eine perfekte Vorhersage indiziert. Jeder Boxplot fasst die Verteilung dieser Vorhersagefehler zusammen: Der farbige Kasten, der Interquartilsabstand (IQR), repräsentiert den Bereich zwischen dem 25. Perzentil (untere Kante) und dem 75. Perzentil (obere Kante) und enthält somit exakt die mittleren 50 % der Datenpunkte. Der durchgezogene schwarze Strich innerhalb des Kastens markiert den Median (50. Perzentil). Die Whiskers (die Linien außerhalb des Kastens) zeigen die restliche Streuung der Daten; Werte, die weit außerhalb dieser Spannweite liegen, werden als einzelne Punkte dargestellt und als Ausreißer interpretiert. Zusätzlich visualisiert der rote Punkt den arithmetischen Mittelwert.

Die gemeinsame Betrachtung von Median und Mittelwert ist für die Modellbewertung von entscheidender Bedeutung, da sie tiefere Einblicke in die Symmetrie und Zuverlässigkeit der Fehlerverteilung gibt. Der Median ist als Zentralwert robust gegenüber Ausreißern

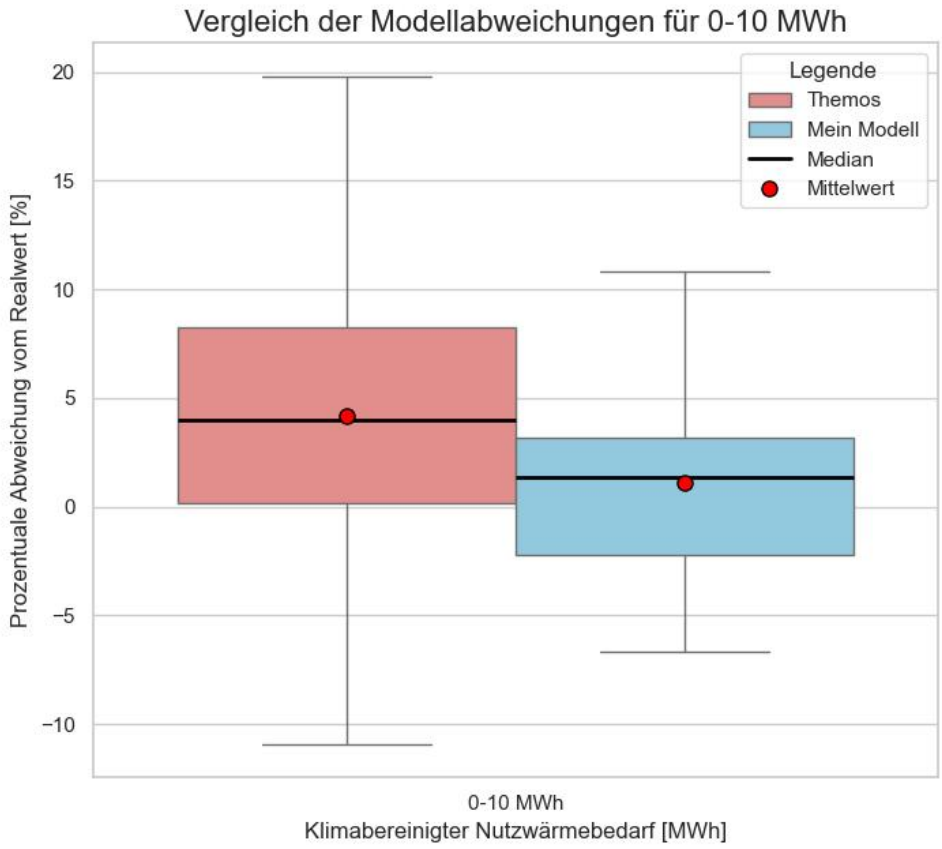

Abbildung 5.2.1: Beispielhafter Boxplot für die Darstellung der Ergebnisse des Trainings. Rot zeigt die prozentuale Abweichung des Thermos-Modells, Blau die des eigenen Modells.

und repräsentiert daher die „typische“ Modellleistung für eine Kategorie. Der Mittelwert hingegen wird von Extremwerten stark beeinflusst. Liegen Mittelwert und Median nah beieinander, deutet dies auf eine symmetrische Fehlerverteilung und eine konsistente, stabile Leistung des Modells hin. Eine große Diskrepanz zwischen den beiden Werten ist jedoch ein wichtiges diagnostisches Signal: Sie zeigt, dass wenige, aber sehr große Vorhersagefehler (Ausreißer) den Durchschnittswert (Mittelwert) stark in ihre Richtung verzerren, obwohl die typische Leistung (Median) möglicherweise gut ist. Dieser Vergleich deckt somit nicht nur die durchschnittliche Genauigkeit auf, sondern auch die Instabilität des Modells, indem er präzise jene Kategorien identifiziert, in denen die Vorhersagen unzuverlässig sind und zu extremen Fehleinschätzungen neigen.

Anhand dieses Vergleichs kann besser beurteilt werden, in welchen Energiebedarfsbe-

reichen das jeweilige Modell im Vergleich zum Referenzmodell besser oder schlechter abschneidet. Wenn sich ein Modell nach dieser Methodik als überlegen erweist, wird dessen Generalisierbarkeit nochmals genauer betrachtet.

Mittels der KFC-Validation kann anschließend, wie in Kapitel 2.3 erwähnt, ermittelt werden, wie stark der RMSE des einmaligen Test- und Trainingsverfahrens vom durch die KFC-Validation gemittelten RMSE abweicht. Je nach Modell können anhand dieser Werte noch Feineinstellungen vorgenommen werden, um die Generalisierbarkeit des Modells zu verbessern.

Nachdem in diesem Kapitel das methodische Vorgehen von der Datenakquise und aufbereitung bis zur Modellevaluation detailliert dargelegt wurde, ist die Basis für die nun folgende Ergebnispräsentation geschaffen. Das nächste Kapitel wird die erzielten Resultate umfassend analysieren, indem die Leistungen der entwickelten Modelle kritisch mit dem Referenzmodell verglichen und anschließend die Strategien zur Optimierung des ausgewählten Modells im Hinblick auf seine Generalisierbarkeit erörtert werden.

6 Auswertung

Die Auswertung erfolgt in der in Kapitel 5 beschriebenen Reihenfolge. Dabei liegen die Datenverteilung sowie die Ergebnisse der beiden ML-Modelle im Fokus. Anschließend wird die Wahl des finalen ML-Modells begründet und dessen Generalisierbarkeit thematisiert.

6.1 Datenverteilung

Nach der Datenaufbereitung sind insgesamt 10.273 Datenpunkte für das Training und die Evaluation der Modelle verfügbar. Wie in Abbildung 6.1.1 zu sehen ist, ist der Datensatz

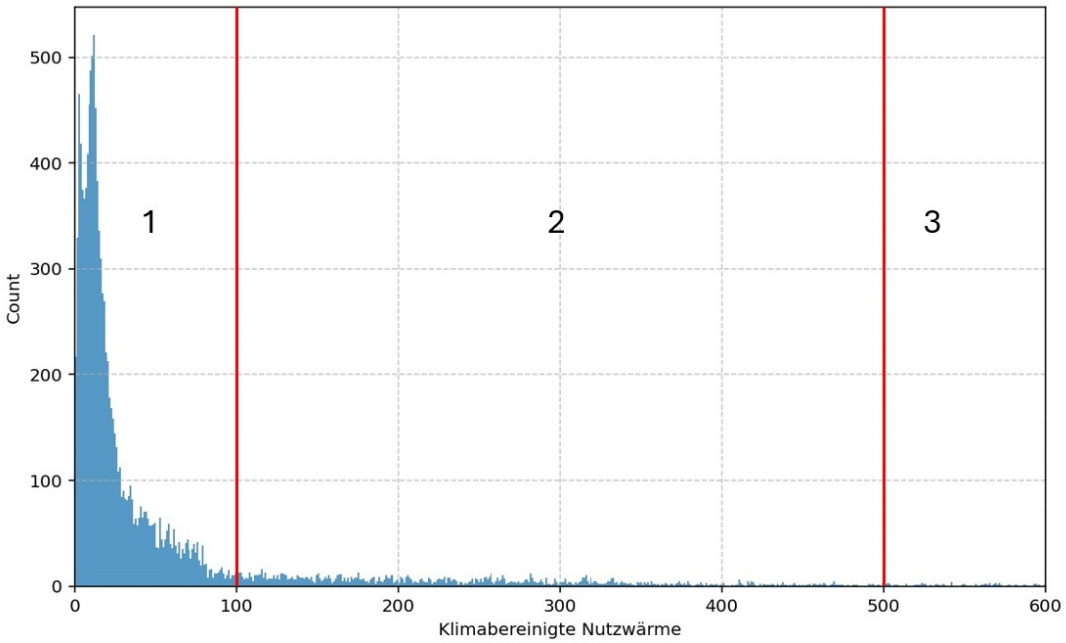

Abbildung 6.1.1: Visualisierung der Datenverteilung bezogen auf den klimabereinigten Nutzwärmebedarf. Bereich 1 (0–100 MWh) umfasst 8.738 Datenpunkte, Bereich 2 (100–500 MWh) 1.423 Datenpunkte und Bereich 3 (>500 MWh) 111 Datenpunkte.

stark unausgeglichen. Dies liegt daran, dass die meisten Gebäude in Dörfern und Städten

Einfamilienhäuser sind. Durch den geringeren Verbrauch dieser Gebäude im Gegensatz zu Mehrfamilienhäusern entsteht diese unausgewogene Verteilung. Um einen Test- und einen Trainingsdatensatz zu erhalten, die beide die Verteilung des Gesamtdatensatzes möglichst gut repräsentieren, wurde der Datensatz nach dem Verbrauch sortiert. Anschließend wurde jeder fünfte Wert für den Testdatensatz verwendet, wodurch eine Aufteilung von 80 % (Training) zu 20 % (Test) realisiert wurde.

6.2 Auswertung der Linearen Regression

Zuerst wurde das LinReg-Modell betrachtet. Die Analyse des Boxplots (Abbildung 6.2.1)

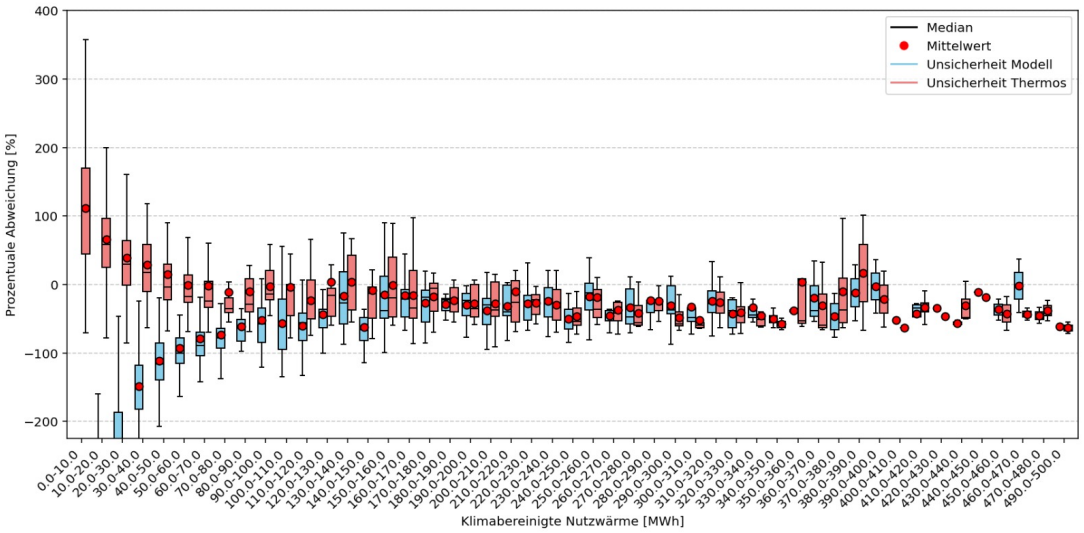

Abbildung 6.2.1: Boxplot-Visualisierung der Ergebnisse des LinReg-Modells (blau) im Vergleich zum Thermos-Modell (rot) für die Spanne von 0–500 MWh.

offenbart eine systematische Schwäche des LinReg-Modells bei der Prognose von Gebäuden mit geringem Wärmebedarf (0–100 MWh). Es ist evident, dass die Vorhersagegenauigkeit mit steigendem Wärmebedarf zunimmt, was auf eine stärkere Linearität im Zusammenhang zwischen den Features und der Zielgröße in höheren Verbrauchsbereichen hindeutet. Besonders kritisch ist die Beobachtung, dass für Gebäude mit geringem Verbrauch häufig negative Wärmebedarfswerte prognostiziert werden, was physikalisch inkonsistent ist. Dieses Verhalten lässt sich direkt auf die Funktionsweise der Methode der kleinsten Quadrate (Ordinary Least Squares) zurückführen, deren Ziel in Gleichung 4.4 definiert wurde. Aufgrund der Quadrierung haben Abweichungen bei Datenpunkten mit hohem Wärmebedarf (Ausreißer nach oben) einen überproportional hohen Einfluss auf

die Verlustfunktion. Um diesen Gesamtfehler zu minimieren, justiert der Algorithmus den Koeffizientenvektor ($\boldsymbol{\beta}$) so, dass die Regressionshyperebene näher an diesen einflussreichen, verbrauchsstarken Gebäuden liegt. Dies geschieht auf Kosten einer signifikanten Unterschätzung im unteren Wertebereich und führt zu einem negativen y-Achsenabschnitt (β_0). Das Modell akzeptiert somit große prozentuale Fehler bei Kleinverbrauchern, um die absoluten quadrierten Fehler bei Großverbrauchern zu reduzieren. Das Ergebnis ist ein, wie in Abbildung 2.4.2 links dargestellt, unterangepasstes ML-Modell.

6.3 Auswertung der Support Vector Regression

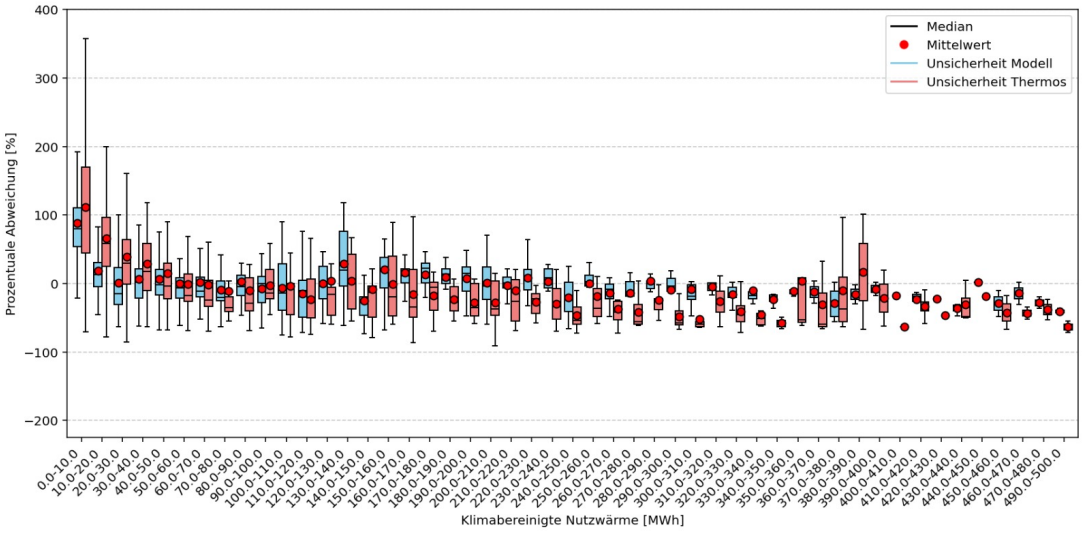

Abbildung 6.3.1: Boxplot-Visualisierung der Ergebnisse des SVR-Modells (blau) im Vergleich zum Thermos-Modell (rot) für die Spanne von 0–500 MWh.

Bei der Brute-Force-Methode zur initialen Parametrisierung des SVR-Modells wurde dem Programm eine Auswahl verschiedener Hyperparameter gegeben. Bei einem Bereich für C von $\{1, 10, 100\}$ und für ϵ von $\{0.1, 0.5, 1\}$ entschied sich der Algorithmus für die jeweils höchsten Parameter ($C = 100$ und $\epsilon = 1$). Bei der Wahl des Kernels erwies sich der RBF-Kernel als die beste Wahl für diesen Anwendungsfall.

Die visuelle Analyse der Modellergebnisse (Abbildung 6.3.1) offenbart die Stärken des SVR-Modells. Im Bereich geringer Nutzwärmebedarfe (0–100 MWh) zeigt das SVR-Modell eine deutlich verbesserte Genauigkeit. Dieses Verhalten, das sich vom LinReg-Modell unterscheidet, liegt am RBF-Kernel und seiner Eigenschaft, auch nicht-lineare Zusammenhänge zu erkennen. Trotz dieser sichtbaren Verbesserung bleibt die durchschnittliche

Vorhersageabweichung für Gebäude im niedrigsten Verbrauchssegment (0–10 MWh) bei annähernd 100 %. Eine Erklärung hierfür liegt in der Nichtberücksichtigung zusätzlicher Wärmequellen. Da nur Energiebedarfe erfasst wurden, die von den Stadtwerken gemeldet wurden, bleiben viele Wärmequellen unberücksichtigt, wie z. B. Kaminöfen oder elektrische Zusatzheizungen. Im mittleren Verbrauchssegment (ca. 60–150 MWh) zeigen beide Modelle eine vergleichbare Leistung. Für höhere Nutzwärmebedarfe (>150 MWh) ist eine Tendenz des Thermos-Modells zur systematischen Unterschätzung festzustellen, während das SVR-Modell hier präzisere Vorhersagen liefert.

Tabelle 6.1: Vergleich der Modellgenauigkeit (RMSE) für verschiedene Methoden.

Methode	**RMSE 0–100**	**RMSE 100–500**	**RMSE >500**	**Gesamt-RMSE**	**Eingabegrößen**
SVR	**14,52**	81,15	330,44	47,56	Umfang, Höhe, Grundfläche, Nutzfläche, Volumen
Lineare Regression	58,83	117,10	315,10	76,87	Umfang, Grundfläche, zusammenhängender Umfang, Außenwandfläche, Volumen, Nutzfläche, Höhe
Thermos (SVR)	16,95	156,67	599,49	86,58	Höhe, Umfang, Volumen, Grundfläche, Außenwandfläche

Tabelle 6.1 zeigt die für jedes Modell optimierten Eingabegrößen, die zu den jeweils besten Ergebnissen führten. Auffällig ist, dass beim SVR-Modell die Hinzunahme des Features „Außenwandfläche“ keine Verbesserung bewirkte, während das LinReg-Modell alle verfügbaren Features nutzte. Die für das Thermos-Modell verwendeten Features wurden der offiziellen Dokumentation entnommen. Im Segment von 0–100 MWh bestätigt die Tabelle die bereits im Boxplot festgestellte Schwäche der LinReg mit einem RMSE von

58,83. Das SVR-Modell zeigt hier mit einem RMSE von 14,52 eine leichte Überlegenheit gegenüber dem Thermos-Modell (RMSE 16,95). Die Tabelle bildet jedoch nicht die visuell erkennbare, höhere Streuung des Thermos-Modells in diesem Segment ab. Besonders hervorzuheben ist die Leistung des SVR-Modells im Bereich von 100–500 MWh, wo es mit einem RMSE von 81,15 als einziges Modell einen Wert unter 100 erreicht. Im höchsten Verbrauchssegment (>500 MWh) erzielt die LinReg eine geringfügig bessere Leistung als das SVR-Modell. Dieses Ergebnis ist jedoch aufgrund der geringen Datenpunktzahl in dieser Kategorie nur begrenzt aussagekräftig.

6.4 Auswahl des ML-Modells

Obwohl der RMSE-Unterschied zwischen dem SVR-Modell (14,52) und dem Thermos-Modell (16,95) im Segment von 0–100 MWh numerisch gering erscheinen mag, ist diese Differenz von 2,43 MWh von erheblicher praktischer Relevanz, da die Mehrheit der Gebäude in diesem Bereich liegt. Eine Genauigkeitsverbesserung hier wirkt sich somit auf einen Großteil des Datensatzes aus. Darüber hinaus zeigt das SVR-Modell eine signifikant geringere Streuung, was auf eine höhere Robustheit hindeutet. Im Segment von 100–500 MWh demonstriert das SVR-Modell seine eindeutige Überlegenheit. Basierend auf diesen Leistungsmerkmalen wird das SVR-Modell für die weitere Bearbeitung ausgewählt.

6.5 Optimierung der Hyperparameter des SVR-Modells

Zur Optimierung der Generalisierbarkeit wurde ein gezieltes Finetuning der Hyperparameter des SVR-Modells durchgeführt. Wie in Kapitel 4.2 erläutert, sind der Kernel-Typ sowie die Parameter C und ϵ entscheidend. Der RBF-Kernel wurde beibehalten. Da der Datensatz stark unausgeglichen ist, wurde die in Kapitel 2.3 erläuterte Stratified KFC-Validation verwendet. Basierend auf der initialen Analyse, die höhere Werte für C und ϵ favorisierte, wurde der Suchraum erweitert: C wurde von 100 bis 1000 und ϵ von 1 bis 10 variiert. Ziel war es, eine Parameterkombination zu finden, die nicht nur einen niedrigen MSE aufweist, sondern auch eine geringe Abweichung zwischen Test-MSE und KFC-MSE, was auf eine gute Generalisierbarkeit hindeutet (siehe Abbildung 6.5.1).

Abbildung 6.5.2 zeigt die Abweichungen detaillierter. Es besteht die Gefahr, dass eine zu starke Regularisierung (hohes C, kleines ϵ) zu einer Überanpassung an die dominante

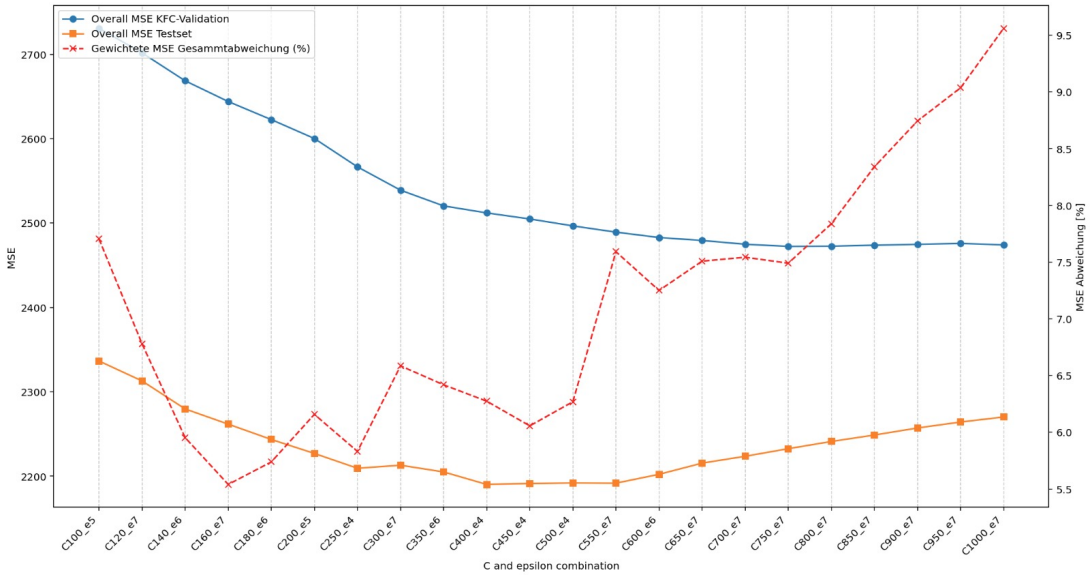

Abbildung 6.5.1: Visualisierung des Verlaufs des MSE für das Testset (Gelb) und des gemittelten MSE der Stratified KFC-Validation (Blau) in Abhängigkeit von C. Die rot gestrichelte Linie zeigt die gewichtete prozentuale Abweichung zwischen diesen beiden Verläufen.

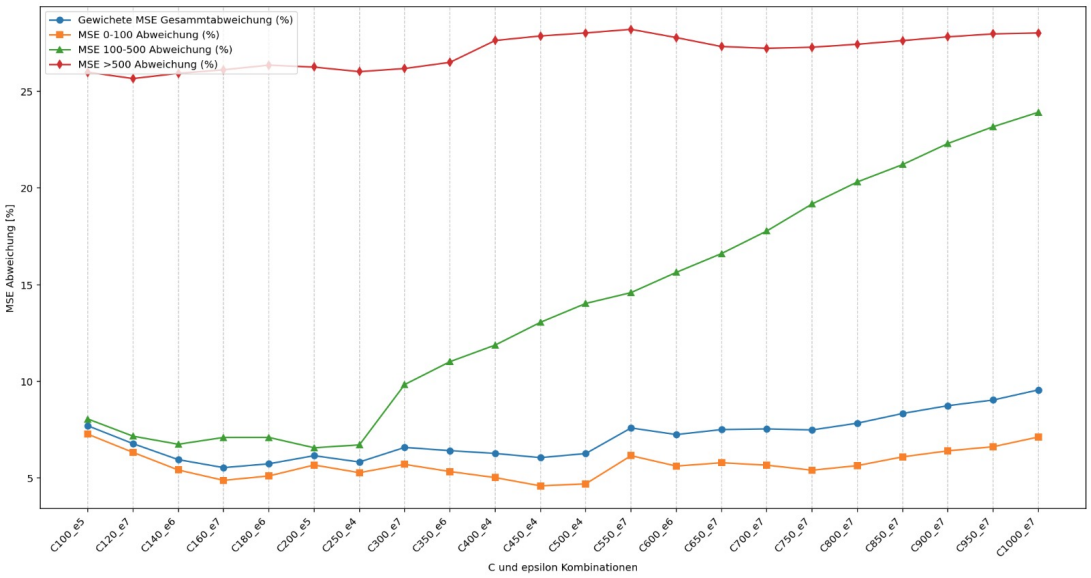

Abbildung 6.5.2: Prozentuale Abweichung zwischen dem Test-MSE und dem KFC-MSE, aufgeschlüsselt nach den drei Verbrauchskategorien und gewichtetem Gesamtergebnis, in Abhängigkeit von C.

Gebäudeklasse (<100 MWh) führt. Es wird daher ein Kompromiss gesucht. Der Gesamt-MSE (Abbildung 6.5.1) sinkt bis zu einer Kombination von $C = 450$ und $\epsilon = 4$. Jedoch

ist die Reduktion von $C = 160$ ($\epsilon = 7$) auf $C = 450$ minimal (MSE sinkt von ca. 2300 auf 2200), während die Generalisierungsabweichung von ca. 5,5 % auf 6,25 % ansteigt. Daher wird die Kombination $C = 160$ mit $\epsilon = 7$ als optimaler Betriebspunkt gewählt, da sie eine gute Leistung bei gleichzeitig höherer Generalisierbarkeit verspricht.

Tabelle 6.2: Vergleich der Modellgenauigkeit (RMSE) nach dem Finetuning.

Merkmal	**SVR ohne Finetuning** ($C = 100$, $\epsilon = 1$)	**SVR mit Finetuning** ($C = 160$, $\epsilon = 7$)	**Thermos**	**Anzahl Gebäude**
RMSE 0–100	14,52	14,18	16,95	8.738
RMSE 100–500	81,15	78,42	156,67	1.423
RMSE >500	330,44	384,41	599,49	111
Gewichtete Gesamtverbesserung [%]	19,35 %	**21,21 %**	0 %	

Die Auswertung in Tabelle 6.2 zeigt, dass durch das Finetuning die Genauigkeit in den Segmenten bis 500 MWh verbessert wurde. Obgleich sich der RMSE im Bereich >500 MWh verschlechtert, ist dies aufgrund der geringen und statistisch nicht aussagekräftigen Datenmenge (111 Gebäude) zu vernachlässigen. Werden alle Bereiche unter Berücksichtigung der Gebäudemenge gewichtet, erzielt das feinabgestimmte SVR-Modell eine um **21,21 %** höhere gewichtete Genauigkeit als das Thermos-Modell. Allein das Finetuning bewirkt eine Genauigkeitssteigerung von **1,86 Prozentpunkten**. Damit wurde eine wesentliche Verbesserung bei der Vorhersage des Heizenergiebedarfs erreicht.

7 Zusammenfassung

Die vorliegende Arbeit befasste sich mit der zentralen Fragestellung, ob auf Basis der von Theta Concepts geschaffenen Datengrundlage ein ML-Modell mit höherer Prognosegenauigkeit als das lizenzierte Thermos-Modell entwickelt werden kann. Zur Beantwortung dieser Frage wurden mittels Literaturrecherche die Lineare Regression (LinReg) und die Support Vector Regression (SVR) als potenziell geeignete Modelle identifiziert. Ein systematischer Vergleich zeigte die Überlegenheit des SVR-Modells, welches anschließend durch ein gezieltes Finetuning der Hyperparameter mittels Stratified KFC-Validation weiter optimiert wurde. Die finalen Ergebnisse belegen den Erfolg dieses Vorgehens eindrücklich: Das optimierte SVR-Modell erzielt im Vergleich zum Thermos-Modell eine um **21,21 %** höhere gewichtete Vorhersagegenauigkeit (siehe Tabelle 6.2). Aus diesem Resultat ergeben sich für die Firma Theta Concepts konkrete Vorteile:

- Eine präzisere Erstellung von Wärmeplänen, wodurch das Risiko einer kostspieligen Über- oder Unterdimensionierung von Projekten signifikant reduziert und die Planungssicherheit erhöht wird.
- Eine nachhaltige Senkung der Betriebskosten, da die Lizenzierung eines externen Drittanbieterprogramms entfällt.

Diese Ergebnisse unterstreichen, dass die systematische Nutzung von archivierten Projektdaten mittels ML einen erheblichen Mehrwert für zukünftige Projekte in Ingenieurbüros schaffen kann. Es ist anzunehmen, dass diese Form der datengestützten Optimierung in Zukunft an Bedeutung gewinnen und der Wert archivierter Daten neu bewertet wird.

8 Anhang

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
100	1.5	rbf	2737.8	204.2	6061.1	157322.6	2334.1	220.3	6567.8	115436.4
100	2.0	rbf	2736.7	203.7	6068.1	157169.4	2334.0	220.2	6561.7	115516.3
100	3.0	rbf	2734.8	203.6	6069.1	156999.3	2333.5	219.3	6552.3	115661.0
100	4.0	rbf	2731.7	203.6	6065.8	156753.4	2334.8	218.2	6556.6	115815.7
100	5.0	rbf	2731.0	203.2	6072.6	156628.2	2336.6	218.0	6561.9	115932.1
100	6.0	rbf	2731.3	203.5	6080.6	156537.9	2339.5	218.4	6569.7	116068.0
100	7.0	rbf	2731.4	204.0	6083.0	156483.3	2342.2	220.2	6567.9	116203.3
100	10.0	rbf	2731.8	208.0	6102.6	155948.5	2346.7	226.5	6549.0	116364.4
120	10.0	rbf	2701.8	207.4	6102.9	153213.3	2320.8	220.0	6560.1	114319.1
120	7.0	rbf	2701.7	202.8	6107.9	153501.3	2312.9	215.6	6545.6	114120.3
120	6.0	rbf	2702.1	202.3	6104.0	153621.7	2310.6	214.7	6546.3	113970.1
120	5.0	rbf	2702.3	202.1	6099.4	153716.9	2308.1	214.3	6536.3	113893.7
120	2.0	rbf	2707.6	202.4	6093.5	154253.8	2304.7	217.9	6526.3	113422.2
120	3.0	rbf	2705.1	202.3	6091.3	154061.7	2303.2	216.4	6527.2	113388.5
120	1.5	rbf	2707.7	202.8	6086.5	154319.2	2305.4	218.1	6532.6	113393.3
120	4.0	rbf	2703.0	202.3	6088.4	153907.3	2305.4	213.6	6536.6	113696.0
140	7.0	rbf	2668.8	201.5	6135.6	150199.5	2280.2	211.7	6553.5	111273.4
140	10.0	rbf	2671.9	206.2	6129.0	150200.0	2283.5	218.1	6564.9	110928.3
140	5.0	rbf	2669.1	201.2	6133.3	150279.4	2278.4	212.5	6534.2	111294.8
140	6.0	rbf	2668.7	201.3	6130.5	150266.6	2279.8	212.2	6544.2	111312.5
140	3.0	rbf	2672.4	201.1	6132.8	150592.7	2275.8	214.0	6510.1	111238.7
140	4.0	rbf	2670.3	201.0	6133.2	150408.7	2277.0	211.2	6532.7	111280.7
140	2.0	rbf	2674.9	201.4	6128.0	150865.5	2277.8	215.8	6509.9	111286.0
140	1.5	rbf	2676.1	202.1	6125.5	150951.1	2277.3	215.7	6503.3	111337.2
160	7.0	rbf	2644.1	201.1	6149.1	147768.8	2261.7	211.0	6585.8	109187.9
160	6.0	rbf	2644.3	200.6	6151.1	147803.1	2261.8	210.4	6580.7	109306.2
160	5.0	rbf	2644.6	200.3	6158.0	147769.1	2261.2	211.2	6564.7	109390.8

Continued on next page

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
160	10.0	rbf	2647.9	205.6	6140.0	147879.6	2262.7	215.6	6593.0	108823.9
160	3.0	rbf	2647.6	200.3	6161.3	148003.1	2258.3	211.5	6533.7	109499.8
160	2.0	rbf	2648.3	200.5	6157.1	148107.3	2255.6	212.4	6514.8	109420.4
160	1.5	rbf	2648.8	201.1	6152.4	148163.1	2254.9	212.9	6505.4	109442.3
160	4.0	rbf	2645.5	200.4	6156.0	147872.5	2259.1	210.5	6549.5	109443.7
180	1.5	rbf	2624.9	200.2	6190.0	145554.2	2239.3	211.7	6536.1	107680.2
180	2.0	rbf	2625.3	199.9	6193.3	145570.6	2241.6	211.8	6546.1	107750.5
180	3.0	rbf	2625.4	199.7	6195.6	145566.1	2244.5	211.5	6563.7	107815.3
180	4.0	rbf	2624.5	199.5	6185.8	145623.3	2244.6	210.0	6584.8	107677.1
180	5.0	rbf	2623.0	199.6	6181.0	145531.5	2244.6	211.0	6594.0	107478.0
180	6.0	rbf	2622.7	200.0	6172.8	145577.1	2243.6	210.2	6611.2	107220.0
180	7.0	rbf	2623.4	200.3	6170.2	145648.0	2243.2	210.2	6622.1	107045.2
180	10.0	rbf	2626.6	205.4	6156.8	145714.7	2245.9	214.1	6631.7	106871.7
200	6.0	rbf	2600.7	199.3	6214.2	143069.3	2226.5	209.5	6646.5	105229.2
200	10.0	rbf	2604.0	205.2	6195.9	143140.6	2229.6	213.6	6665.1	104955.0
200	7.0	rbf	2601.1	199.9	6208.7	143121.6	2227.4	210.5	6660.9	105045.5
200	5.0	rbf	2600.3	199.1	6216.9	143012.7	2227.0	210.4	6625.1	105475.8
200	2.0	rbf	2602.3	199.5	6240.1	142879.6	2226.3	211.3	6587.9	105820.8
200	3.0	rbf	2602.0	199.0	6237.2	142923.4	2226.6	211.0	6593.9	105791.4
200	4.0	rbf	2601.2	199.1	6223.0	143024.5	2226.6	209.4	6619.1	105606.1
200	1.5	rbf	2602.2	199.8	6238.5	142858.0	2225.0	211.7	6580.9	105763.1
250	1.5	rbf	2569.6	198.6	6312.1	138987.1	2207.4	211.2	6686.1	102791.9
250	2.0	rbf	2568.3	198.7	6302.9	138979.2	2208.1	211.3	6691.1	102790.5
250	3.0	rbf	2567.4	197.9	6298.9	139014.1	2208.5	210.8	6688.8	102888.6
250	4.0	rbf	2566.7	198.0	6285.3	139114.4	2209.4	208.4	6707.4	102927.3
250	5.0	rbf	2566.7	198.4	6271.4	139264.4	2212.3	209.2	6725.8	102903.9
250	6.0	rbf	2567.1	198.8	6264.7	139347.7	2215.4	209.3	6746.7	102911.2
250	7.0	rbf	2567.6	199.0	6262.8	139406.8	2217.6	209.8	6752.7	103000.5
250	10.0	rbf	2570.4	204.4	6237.2	139560.9	2222.4	215.2	6761.3	102909.8
300	7.0	rbf	2539.0	198.8	6247.1	136986.1	2213.0	210.1	6861.6	101131.8
300	6.0	rbf	2539.5	198.2	6254.2	136991.7	2210.3	209.8	6844.4	101122.3
300	5.0	rbf	2539.5	197.8	6261.9	136916.8	2207.2	210.1	6827.7	101028.0
300	10.0	rbf	2541.1	204.2	6233.7	136917.1	2221.4	217.3	6880.4	101096.2
300	3.0	rbf	2541.6	197.5	6288.6	136804.6	2203.7	211.8	6794.8	100996.1
300	2.0	rbf	2543.4	198.3	6295.3	136815.7	2203.5	211.4	6796.8	100986.4
300	1.5	rbf	2544.6	198.4	6300.6	136850.4	2202.8	211.2	6796.5	100940.8
300	4.0	rbf	2540.6	197.9	6270.1	136906.5	2204.8	209.6	6820.1	100942.3

Continued on next page

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
350	7.0	rbf	2520.5	198.5	6254.8	135209.7	2209.0	210.1	6966.5	99401.6
350	6.0	rbf	2520.4	197.9	6262.0	135154.3	2205.1	208.5	6951.9	99348.1
350	5.0	rbf	2520.5	197.5	6270.9	135074.0	2200.2	208.6	6928.0	99196.8
350	10.0	rbf	2522.3	204.2	6227.0	135278.0	2212.5	217.5	6954.8	99290.8
350	3.0	rbf	2522.0	197.2	6291.4	134978.8	2197.2	210.9	6893.2	99185.6
350	2.0	rbf	2523.6	198.1	6293.0	135035.0	2194.9	211.5	6877.3	99128.8
350	1.5	rbf	2525.1	198.2	6297.4	135109.4	2195.3	211.4	6878.8	99147.5
350	4.0	rbf	2521.2	197.6	6274.8	135088.4	2199.1	208.3	6923.2	99184.9
400	1.5	rbf	2513.9	198.2	6288.2	134185.4	2190.4	211.8	6977.7	97385.7
400	2.0	rbf	2513.3	198.2	6284.6	134180.5	2190.5	211.4	6980.7	97379.9
400	3.0	rbf	2512.8	197.6	6285.6	134177.8	2187.9	210.5	6978.3	97247.3
400	4.0	rbf	2512.1	197.6	6273.6	134263.2	2190.3	207.5	7018.8	97176.1
400	5.0	rbf	2513.3	197.4	6277.2	134340.0	2189.8	206.9	7026.0	97087.1
400	6.0	rbf	2513.4	197.7	6267.5	134450.2	2195.1	207.2	7054.8	97185.8
400	7.0	rbf	2512.8	198.4	6256.9	134473.2	2196.4	209.6	7056.7	97088.9
400	10.0	rbf	2514.4	204.2	6218.6	134650.0	2194.9	217.7	7025.2	96715.7
450	6.0	rbf	2507.2	197.7	6270.4	133843.9	2190.5	208.2	7091.2	96199.1
450	7.0	rbf	2506.6	198.4	6260.6	133858.2	2190.4	210.0	7086.8	96113.1
450	5.0	rbf	2506.7	197.3	6280.0	133705.1	2190.3	206.0	7099.5	96259.5
450	10.0	rbf	2508.8	203.9	6223.9	134091.0	2188.8	217.7	7050.9	95818.5
450	3.0	rbf	2505.5	197.6	6281.3	133555.4	2190.1	210.1	7069.9	96299.6
450	2.0	rbf	2506.2	198.5	6277.4	133601.6	2192.2	211.5	7086.6	96163.9
450	1.5	rbf	2506.1	198.3	6277.3	133606.4	2191.1	211.6	7075.8	96192.8
450	4.0	rbf	2504.9	197.6	6275.5	133575.2	2191.4	206.7	7094.7	96365.8
500	7.0	rbf	2496.9	198.3	6269.5	132859.4	2188.2	210.1	7127.3	95370.9
500	6.0	rbf	2498.0	197.7	6280.0	132876.3	2191.5	209.4	7149.3	95449.9
500	5.0	rbf	2497.8	197.1	6283.6	132851.6	2192.0	206.4	7160.6	95593.9
500	10.0	rbf	2497.3	204.0	6235.8	132872.8	2186.9	218.1	7096.7	95012.4
500	3.0	rbf	2498.5	197.5	6286.7	132851.1	2194.3	209.5	7158.2	95590.8
500	2.0	rbf	2499.4	198.5	6284.4	132878.7	2195.6	210.2	7176.3	95426.0
500	1.5	rbf	2499.4	198.3	6286.8	132861.0	2195.7	211.1	7174.9	95377.8
500	4.0	rbf	2496.7	197.6	6276.4	132807.1	2192.0	206.9	7156.9	95603.1
550	1.5	rbf	2492.9	198.0	6301.8	132100.2	2203.6	209.5	7285.6	94809.7
550	2.0	rbf	2492.8	198.2	6298.6	132108.8	2203.0	209.4	7278.9	94842.8
550	3.0	rbf	2491.6	197.5	6288.7	132185.5	2200.6	208.9	7258.6	94928.4
550	4.0	rbf	2489.5	197.5	6280.0	132109.9	2198.7	207.6	7248.9	94973.6
550	5.0	rbf	2489.6	197.1	6286.2	132068.6	2194.1	206.8	7222.6	94951.3

Continued on next page

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
550	6.0	rbf	2489.6	197.7	6283.2	132063.8	2194.3	209.1	7215.5	94886.1
550	7.0	rbf	2489.3	198.3	6278.2	132050.7	2191.8	210.5	7194.0	94812.0
550	10.0	rbf	2490.1	204.3	6241.5	132111.7	2191.2	217.9	7170.4	94477.0
600	6.0	rbf	2483.0	197.8	6292.5	131318.6	2202.2	208.9	7276.4	94847.2
600	10.0	rbf	2486.1	204.4	6254.3	131579.3	2199.7	218.2	7235.7	94401.5
600	7.0	rbf	2483.4	198.8	6289.1	131323.1	2201.5	210.4	7266.6	94790.1
600	5.0	rbf	2483.7	197.5	6301.6	131304.6	2202.1	207.4	7288.0	94805.8
600	4.0	rbf	2484.7	197.6	6299.8	131399.9	2206.5	207.4	7320.9	94794.4
600	3.0	rbf	2486.6	197.6	6308.3	131470.3	2208.8	208.6	7334.3	94733.7
600	2.0	rbf	2487.6	198.0	6320.4	131369.9	2212.3	208.1	7368.2	94668.5
600	1.5	rbf	2488.2	197.9	6320.7	131430.0	2212.8	208.9	7371.1	94607.8
650	1.5	rbf	2485.3	198.0	6326.1	131085.4	2226.0	208.6	7434.7	95038.5
650	2.0	rbf	2484.2	198.0	6325.4	130997.0	2224.1	207.8	7423.9	95072.3
650	3.0	rbf	2482.7	197.7	6317.3	130989.1	2221.7	208.7	7395.3	95138.5
650	4.0	rbf	2481.5	197.8	6309.8	130962.9	2219.3	208.3	7379.3	95163.3
650	5.0	rbf	2479.8	197.7	6310.4	130817.8	2216.5	208.0	7357.4	95204.8
650	6.0	rbf	2479.6	198.1	6300.7	130884.0	2215.2	208.6	7345.7	95181.5
650	7.0	rbf	2479.4	199.0	6294.4	130879.4	2215.5	210.5	7339.9	95138.0
650	10.0	rbf	2482.3	204.4	6260.4	131154.8	2210.6	218.8	7288.2	94683.4
700	10.0	rbf	2478.1	204.5	6260.7	130750.5	2220.4	219.3	7365.0	94571.1
700	7.0	rbf	2475.0	198.8	6296.8	130454.5	2223.6	210.1	7416.1	94944.0
700	6.0	rbf	2476.3	198.2	6308.0	130481.3	2223.7	208.0	7429.8	94937.2
700	5.0	rbf	2477.3	197.8	6317.2	130489.9	2226.2	208.2	7443.0	94983.1
700	1.5	rbf	2482.7	198.1	6332.0	130766.4	2236.2	207.9	7509.8	95076.5
700	3.0	rbf	2479.9	197.8	6325.0	130627.1	2231.1	208.4	7473.0	95037.0
700	2.0	rbf	2481.2	198.1	6329.5	130667.0	2233.7	208.1	7489.4	95094.7
700	4.0	rbf	2479.3	197.9	6318.9	130636.4	2227.4	208.2	7451.6	94983.8
750	6.0	rbf	2474.4	198.0	6309.1	130315.7	2234.4	207.2	7523.5	94784.6
750	7.0	rbf	2472.4	198.8	6293.7	130260.8	2232.6	209.6	7500.3	94733.0
750	10.0	rbf	2475.1	204.5	6257.2	130520.6	2229.6	220.3	7435.3	94435.4
750	5.0	rbf	2476.0	197.9	6316.8	130365.9	2236.9	207.2	7540.8	94794.2
750	2.0	rbf	2479.6	198.2	6333.0	130467.2	2242.9	208.3	7567.3	94930.0
750	3.0	rbf	2478.5	197.8	6331.3	130422.6	2238.5	208.3	7540.3	94861.9
750	1.5	rbf	2481.1	198.2	6334.9	130574.4	2245.3	208.1	7585.3	94932.5
750	4.0	rbf	2478.2	197.9	6326.5	130448.1	2236.6	207.7	7534.8	94803.6
800	1.5	rbf	2480.8	198.3	6333.3	130559.3	2252.5	207.6	7652.7	94769.3
800	2.0	rbf	2479.3	198.3	6331.5	130459.7	2251.1	208.2	7639.8	94765.5

Continued on next page

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
800	3.0	rbf	2477.8	197.8	6326.9	130410.7	2246.8	208.3	7613.9	94686.4
800	4.0	rbf	2477.0	197.9	6320.5	130407.0	2246.9	207.2	7626.3	94629.3
800	5.0	rbf	2475.7	197.8	6316.5	130352.5	2247.4	206.5	7634.6	94621.2
800	6.0	rbf	2474.1	197.7	6310.5	130280.4	2244.4	207.6	7612.1	94543.8
800	7.0	rbf	2472.6	198.8	6297.0	130239.5	2241.2	210.0	7576.0	94516.4
800	10.0	rbf	2474.8	204.4	6259.8	130474.5	2236.7	221.1	7496.6	94248.4
850	10.0	rbf	2476.3	204.2	6277.0	130408.0	2244.4	221.7	7564.2	94040.0
850	7.0	rbf	2473.9	198.6	6309.5	130208.5	2248.8	210.7	7647.7	94247.3
850	6.0	rbf	2474.8	197.7	6318.6	130255.3	2253.3	208.4	7694.2	94245.8
850	5.0	rbf	2476.8	197.6	6326.6	130346.3	2257.1	207.0	7717.4	94411.4
850	1.5	rbf	2479.1	198.4	6329.4	130453.9	2260.0	206.6	7728.2	94580.6
850	3.0	rbf	2477.5	197.8	6329.1	130361.2	2258.7	208.6	7711.8	94506.1
850	2.0	rbf	2478.0	198.2	6332.2	130329.8	2258.6	207.6	7710.5	94600.2
850	4.0	rbf	2477.1	197.8	6322.6	130401.1	2257.7	206.5	7720.4	94472.7
900	10.0	rbf	2477.4	204.1	6285.4	130400.7	2251.3	222.0	7623.4	93890.2
900	7.0	rbf	2474.8	198.6	6313.5	130249.1	2257.2	211.3	7721.4	94024.9
900	6.0	rbf	2476.0	197.5	6323.7	130303.0	2261.2	209.4	7756.8	94093.5
900	5.0	rbf	2477.1	197.6	6331.1	130302.4	2267.0	207.6	7796.9	94255.4
900	3.0	rbf	2478.2	197.7	6331.9	130388.1	2270.2	208.4	7806.1	94369.9
900	2.0	rbf	2478.2	198.3	6333.7	130320.7	2267.7	207.2	7792.7	94406.8
900	1.5	rbf	2479.0	198.4	6335.0	130372.0	2270.2	206.6	7814.1	94416.6
900	4.0	rbf	2477.4	197.7	6325.5	130405.8	2268.4	206.2	7807.7	94354.4
950	10.0	rbf	2478.3	204.2	6292.7	130390.7	2257.3	222.3	7677.5	93729.4
950	6.0	rbf	2477.4	197.6	6331.2	130331.1	2268.5	210.4	7815.4	93937.5
950	5.0	rbf	2478.3	197.7	6336.1	130347.6	2274.1	208.3	7857.7	94079.8
950	7.0	rbf	2475.9	198.6	6320.4	130254.6	2264.3	211.8	7784.6	93830.0
950	3.0	rbf	2479.3	197.8	6336.7	130430.1	2277.3	208.3	7872.4	94188.5
950	2.0	rbf	2480.1	198.4	6338.7	130420.7	2275.7	207.7	7866.1	94166.7
950	1.5	rbf	2480.6	198.5	6341.2	130431.3	2278.1	207.1	7885.8	94185.1
950	4.0	rbf	2478.5	197.7	6330.7	130437.8	2275.0	206.8	7867.1	94163.5
1000	7.0	rbf	2474.2	198.6	6325.8	130025.5	2270.2	212.8	7838.6	93604.2
1000	1.5	rbf	2479.8	198.6	6347.2	130275.0	2285.4	208.0	7948.3	93983.3
1000	2.0	rbf	2479.2	198.5	6345.4	130253.2	2282.8	208.4	7929.6	93950.3
1000	3.0	rbf	2477.9	197.9	6340.4	130237.9	2283.4	209.3	7927.0	93963.7
1000	4.0	rbf	2476.8	197.7	6333.7	130238.4	2281.7	207.5	7927.0	93947.5
1000	5.0	rbf	2476.3	197.9	6338.4	130121.0	2280.8	209.3	7915.3	93872.3
1000	6.0	rbf	2475.2	197.7	6335.0	130070.1	2274.6	211.3	7870.2	93721.6

Continued on next page

C	epsilon	kernel	overall mse	mse 0-100	mse 100-500	mse >500	overall mse test	mse 0-100 test	mse 100-500 test	mse >500 test
1000	10.0	rbf	2477.1	204.3	6294.9	130237.3	2262.1	222.9	7723.7	93537.5

Literaturverzeichnis

[1] IPCC (2021): „Klimawandel Naturwissenschaftliche Grundlagen“.

[2] eurostat (2025): *Energy consumption in households. Statistics Explained.* Quelle: URL: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Energy_consumption_in_households.

[3] European Enviroment Agency (2023): *Greenhouse gas emissions from energy use in buildings in Europe*, Quelle: URL: https://www.eea.europa.eu/en/analysis/maps-and-charts/greenhouse-gas-emissions-from-energy-6?utm_source=chatgpt.com&activeTab=570bee2d-1316-48cf-adde-4b640f92119b.

[4] Gesetz für die Wärmeplanung und zur Dekarbonisierung der Wärmenetze (2024): *Wärmeplanungsgesetz: WPG.*

[5] Christopher M. Bishop (2006): *Pattern Recognition and Machine Learning.*

[6] Andreas François Vermeulen (2020): *Industrial Machine Learning: Using Artificial Intelligence As a Transformational Disruptor*, Quelle: URL: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5988135.

[7] Alexander Jung (2024): *Maschinelles Lernen: Die Grundlagen.*

[8] Nils Gerrit Brinkmann (2019): *Texterkennung auf handgeschriebenen Dokumenten mit Long Short-Term Memory Networks.*

[9] Mathias Trabs u. a. (2021): *Statistik und maschinelles Lernen: Eine mathematische Einführung in klassische und moderne Methoden.*

[10] Trevor Hastie, Robert Tibshirani und Jerome H. Friedman (2009): *The elements of statistical learning: Data mining, inference, and prediction.*

[11] Vinod Chugani (2025): *Ein umfassender Leitfaden zur K-Fold Cross Validation.*

[12] Agnieszka Mikołajczyk Bareła und Michal Grochowski (2023): *A Survey on Bias in Machine Learning Research.*

[13] Pritha Bhandari (2023): *Random vs. Systematic Error | Definition & Examples.*

[14] Dewang Nautiyal (11.03.2024): *ML | Unteranpassung und Überanpassung*, Quelle: URL: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/.

[15] Radiša Ž. Jovanović, Aleksandra A. Sretenović und Branislav D. Živković (2015): „Ensemble of various neural networks for prediction of heating energy consumption“.

[16] Zeyu Wang und Ravi S. Srinivasan (2016): „A review of artificial intelligence based building energy use prediction_ Contrasting the capabilities of single and ensemble prediction models“.

[17] Thermos (2021): „validation-study-thermos“.

[18] Siddhesh Bangar (2023): *Understanding Support Vector Regression: A Guide to Machine Learning's Powerful Tool*.

[19] Dover McGraw-Hill (1959): *A Source Book in Mathematics*.

[20] Vladimir N. Vapnik (2000): *The Nature of Statistical Learning Theory*.

[21] Deutsches Institut für Normung e. V. (2021): *DIN 277*.