Hochschule München
University of Applied Sciences

Fakultät Für Informatik und Mathematik
Stochastic Engineering in Business and Finance

Master-Thesis

Interpretation linearer Modelle mit SHAP

Interpretation of linear models with SHAP

Betreuer: Prof. Dr. Andreas Zielke

Eingereicht von:
Simon Symhoven, 49651418
Boschetsriederstraße 59A, D-81379 München
simon.symhoven@hm.edu

Eingereicht am:
München, den 16. März 2024

Abstract

In der vorliegenden Masterarbeit wird die Interpretation linearer Modelle mittels SHAP (engl. *SHapley Additive exPlanations*) untersucht, einem Verfahren, das seine Grundlagen in den Shapley-Werten der kooperativen Spieltheorie findet. Die Untersuchung beginnt mit einem Rückblick auf die historische Entwicklung der Shapley-Werte und setzt sich mit einer detaillierten Herleitung sowie einer Analyse der zugrundeliegenden axiomatischen Prinzipien fort. Es wird dargelegt, wie SHAP auf Basis der Shapley-Werte konzipiert wurde, um Beiträge der einzelnen Merkmale zur Vorhersagegenauigkeit eines Modells zu quantifizieren. Die Anwendung des Konzepts auf einen Datensatz aus der Praxis, bearbeitet mit dem Python-Paket shap, verdeutlicht die Handhabung und praktische Relevanz des Ansatzes. Den Schlussstein der Arbeit bildet ein Abgleich der gewonnenen Einsichten durch SHAP mit traditionellen Methoden zur Bestimmung der Relevanz von Modellmerkmalen. Zudem wird eine kritische Betrachtung der Grenzen und Herausforderungen, die mit SHAP einhergehen, präsentiert. Ein wesentliches Ergebnis der vorliegenden Arbeit ist die Erkenntnis, dass SHAP eine tiefere und nuanciertere Analyse der Modellvorhersagen ermöglicht, indem es nicht nur globale, sondern auch individuelle Merkmalsbeiträge hervorhebt. Diese Fähigkeit, sowohl lokale als auch globale Interpretationen zu liefern, unterscheidet SHAP wesentlich von traditionellen Methoden zur Bestimmung der Merkmalsrelevanz.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

1. Einleitung

In einer Ära, in der datengetriebene Entscheidungsfindung und automatisierte Modelle zunehmend an Bedeutung gewinnen, wird die Fähigkeit der Erklärung und Interpretation dieser Modelle immer wichtiger. Obwohl maschinelles Lernen und künstliche Intelligenz beachtliche Fortschritte erzielt haben, ist es nicht allein die Komplexität der Modelle, die Verständnisschwierigkeiten bereitet, sondern vor allem die Transparenz ihrer Entscheidungsprozesse. Während der grundlegende Aufbau eines Modells, wie etwa eines neuronalen Netzes mit einer überschaubaren Anzahl an Neuronen, durchaus verständlich sein kann, bleibt die spezifische Art und Weise, wie das trainierte Modell Daten verarbeitet und zu Vorhersagen gelangt, oft undurchsichtig und schwer nachvollziehbar [Mol23, S. 4f]. Selbst einfache Modelle, wie eine lineare Regression, können durch Transformationen der Eingabemerkmale oder der Zielvariable schwer zu interpretieren sein [Mol23, S. 6]. Warum ein bestimmtes Modell eine entsprechende Entscheidung getroffen hat oder wie sich die einzelnen Merkmale auf die Vorhersagen auswirken, wird zu einer zentralen Frage für das Verständnis der zugrundeliegenden Mechanismen. Shapley-Werte bieten eine Möglichkeit, die „Black Box“ der Modelle zu öffnen und Einblicke in die Modellentscheidung zu gewinnen [Mol22, S. 215f].

Diese Masterarbeit widmet sich einer tiefgehenden Untersuchung der Shapley-Werte und erforscht ihr Potenzial für die Interpretation von Machine Learning-Modellen, insbesondere linearer Modelle, sowie ihre Anwendbarkeit auf reale Datensätze.

Die Arbeit beginnt mit einer Einführung in die Shapley-Werte, indem ihre historischen Ursprünge und ihre Verbindung zur kooperativen Spieltheorie beleuchtet werden. Die theoretischen Grundlagen dieser Werte werden ergründet und ihre Bedeutung für die Interpretation von Modellen untersucht.

Anschließend wird der Fokus auf die Anpassung und Erweiterung der Shapley-Werte für maschinelles Lernen gelegt. Hierbei wird betrachtet, wie Shapley-Werte modifiziert werden können, um tiefere Einsichten in die Gewichtung und Bedeutung einzelner Merkmale innerhalb komplexer Machine Learning-Modelle zu ermöglichen. Dabei wird ein Vergleich mit bestehenden Methoden, wie der Permutation der Merkmalrelevanz, gezogen, wodurch die Einzigartigkeit und der Mehrwert der Shapley-Werte hervorgehoben wird.

Durch die Analyse eines realen Datensatzes wird die Methodik in der Praxis demonstriert. Die Fallstudie zur Vorhersage der Druckfestigkeit von Beton bietet dabei

ein konkretes Beispiel, anhand dessen die Nuancen und die Tiefe der durch SHAP ermöglichten Analysen veranschaulicht werden.

Abschließend werden die gewonnenen Erkenntnisse zusammengefasst und reflektiert. Es wird speziell die Rolle und das Potenzial der Shapley-Werte in diesem Kontext beleuchtet.

1.1. Der Ursprung der Shapley-Werte in der kooperativen Spieltheorie

Der Ursprung der Shapley-Werte liegt in der kooperativen Spieltheorie, einem fundamentalen Zweig der Spieltheorie. Dieser Bereich beschäftigt sich mit der Analyse von Situationen, in denen Akteure zusammenarbeiten, um gemeinsame Ziele zu erreichen. Zentrales Anliegen ist dabei die gerechte Verteilung der resultierenden Gewinne unter den Akteuren. Ein Schlüsselkonzept dieser Theorie ist die sogenannte charakteristische Funktion, welche die Bewertung der Gewinnverteilung einer Koalition von Akteuren ermöglicht [Mol23, S. 12].

Die Shapley-Werte, entwickelt von Lloyd Shapley in den 1950er Jahren, bieten einen methodischen Ansatz, um den individuellen Beitrag eines jeden Akteurs zur kooperativen Zusammenarbeit gerecht zu bewerten[1]. Dies geschieht durch die Durchschnittsbewertung der Beiträge über sämtliche mögliche Koalitionen hinweg. Diese Methode erweist sich als äußerst nützlich, um eine gerechte und rationale Verteilung von Gewinnen in vielfältigen Szenarien zu ermöglichen, sei es in wirtschaftlichen Verhandlungen oder der Aufteilung von Ressourcen [RWB+22, S. 5572].

Das Verständnis der kooperativen Spieltheorie und ihrer Anwendung in Form der Shapley-Werte ermöglicht es, dieses theoretische Konzept auf den Bereich des maschinellen Lernens zu übertragen. Im Kontext des maschinellen Lernens wird das Spiel zur Modellvorhersage einer konkreten Beobachtung. Die Spieler werden als Eingabemerkmale aufgefasst und bilden eine Koalition. Der Gewinn resultiert aus der Differenz einer Vorhersage und der durchschnittlichen Vorhersage des Modells über alle Beobachtungen [Mol22, S. 215]. Die charakteristische Funktion beschreibt die Gesamtleistung, die eine Kombination von Merkmalen auf die Vorhersage des Modells ausübt [RWB+22, S. 5572].

1.2. Shapley-Werte, SHAP, SHAP-Werte und shap

Zur Verdeutlichung und Abgrenzung der verschiedenen, jedoch verwandten Begrifflichkeiten, die im Kontext dieser Arbeit Verwendung finden, ist eine kurze Einordnung essenziell.

Beginnend mit den Shapley-Werten, entstammt dieser Begriff der kooperativen Spieltheorie und beschreibt eine Methode, um den fairen Beitrag eines Spielers zu der Gesamtauszahlung eines kooperativen Spiels zu bestimmen.

SHAP (SHapley Additive exPlanations) ist ein Interpretationsframework für maschinelles Lernen, das Shapley-Werte verwendet, um den Beitrag jedes Merkmals zur Modellvorhersage zu quantifizieren. Dieses Konzept wurde erstmals von Lundberg und Lee 2017 eingeführt und ermöglicht die Erklärung komplexer Modelle und die Interpretation von Vorhersagen [LL17, S. 1].

[1] Erstmalige Veröffentlichung: „A Value for n-Person Games“, [Sha53, S. 307-318]

Bei den SHAP-Werten handelt es sich um die konkreten quantitativen Beiträge der einzelnen Merkmale zu einer bestimmten Vorhersage, berechnet basierend auf dem SHAP-Framework.

Das Python-Paket shap schließlich ist eine Implementierung, die es praktikabel macht, SHAP-Werte in der Anwendung zu berechnen und zu visualisieren. Es stellt eine reiche Auswahl an Werkzeugen zur Verfügung, um diese Werte und ihre Auswirkungen zu interpretieren [Mol23, S. 14].

2. Theorie der Shapley-Werte

Durch die Verwendung eines praktischen Beispiels – der Aufteilung eines Preisgeldes aus einem Designwettbewerb unter den Gewinnern – wird zunächst eine intuitive Einführung in das Konzept gegeben. Anschließend wird die formale Definition der Shapley-Werte erläutert, um die theoretischen Grundlagen für ihre Berechnung und Anwendung zu legen.

2.1. Wie lässt sich der Gewinn gerecht aufteilen?

Angenommen, drei Teilnehmer, Anna, Ben und Carla, haben als Team kooperiert und den ersten Platz bei einem Designwettbewerb belegt[2]. Dieser Erfolg führt zu einem Gesamtgewinn von 1000 €. Das Preisgeld für den zweiten Platz beträgt 750 € und 500 € für den dritten Platz. Die Herausforderung besteht nun darin, den Gewinn auf eine Weise zu verteilen, die den individuellen Beitrag jedes Teilnehmers zur Erzielung des ersten Platzes gerecht widerspiegelt.

Die Situation wird komplizierter, wenn man bedenkt, dass jeder Teilnehmer unterschiedlich zu dem Erfolg beigetragen hat und ihre individuellen Leistungen auch zu verschiedenen Ausgängen geführt hätten, wenn sie alleine oder in anderen Teilkonstellationen angetreten wären.

Um eine faire Aufteilung des Preisgeldes zu erreichen, betrachten wir die hypothetischen Gewinne, die Anna, Ben und Carla erzielt hätten, wenn sie in unterschiedlichen Konstellationen am Wettbewerb teilgenommen hätten. Tabelle 1 zeigt die gegebene Gewinnverteilung der verschiedenen Koalitionen. Die Koalition $\emptyset$ entspricht dabei der leeren Koalition – der Nichtteilnahme an dem Wettbewerb.

Zur Berechnung der Shapley-Werte ist es erforderlich, den marginalen Beitrag jedes Spielers zu erfassen. Marginalbeiträge in der Spieltheorie, und speziell im Kontext der Shapley-Werte, sind die zusätzlichen Beiträge, die ein Spieler (Teilnehmer) zum Gesamtgewinn einer Koalition beiträgt, wenn er dieser beitritt. Die Berechnung des marginalen Beitrags eines Teilnehmers erfolgt, indem man den Wert der Koalition ohne diesen Teilnehmer vom Wert der Koalition mit dem Teilnehmer subtrahiert [Mol23, S. 18].

In diesem Beispiel mit Anna, Ben und Carla, die an einem Designwettbewerb teilnehmen, ist der marginale Beitrag von Anna zur Koalition von {Ben} der zusätzliche

[2] In Anlehnung an das Beispiel aus Kapitel 4.1 „Who's going to pay for that taxi?“ [Mol23, S.17-20]. Daten sind hypothetisch zu Illustrationszwecken frei erfunden.

Tabelle 1.: Potenzielle Gewinne für verschiedene Teilnehmerkonstellationen im Designwettbewerb.

Koalition	Gewinn	Bemerkung
∅	0 €	Keine Teilnahme
{Anna}	500 €	3. Platz als Einzelteilnehmerin
{Ben}	750 €	2. Platz als Einzelteilnehmer
{Carla}	0 €	Kein Gewinn als Einzelteilnehmerin
{Anna, Ben}	750 €	2. Platz als Team ohne Carla
{Anna, Carla}	750 €	2. Platz als Team ohne Ben
{Ben, Carla}	500 €	3. Platz als Team ohne Anna
{Anna, Ben, Carla}	1000 €	1. Platz als Gesamtteam

Quelle: Eigene Darstellung.

Wert, den Anna einbringt, wenn sie sich Ben anschließt, ausgehend von Bens individuellem Gewinn.

Tabelle 2.: Marginalbeiträge der einzelnen Teilnehmer zu den möglichen Koalitionen.

Teilnehmer	Zur Koalition	Gewinn vorher	Gewinn nachher	Marginalbeitrag
Anna	∅	0 €	500 €	500 €
Anna	{Ben}	750 €	750 €	0 €
Anna	{Carla}	0 €	750 €	750 €
Anna	{Ben, Carla}	500 €	1000 €	500 €
Ben	∅	0 €	750 €	750 €
Ben	{Anna}	500 €	750 €	250 €
Ben	{Carla}	0 €	500 €	500 €
Ben	{Anna, Carla}	750 €	1000 €	250 €
Carla	∅	0 €	0 €	0 €
Carla	{Anna}	500 €	750 €	250 €
Carla	{Ben}	750 €	500 €	-250 €
Carla	{Anna, Ben}	750 €	1000 €	250 €

Quelle: Eigene Darstellung.

Tabelle 2 illustriert den Gewinn jeder möglichen Koalition ohne den betrachteten Spieler und den neuen Gesamtgewinn, sobald dieser Spieler der Koalition beitritt. Der marginale Beitrag jedes Spielers wird als Differenz zwischen diesen beiden Werten berechnet und gibt Aufschluss über den individuellen Wertbeitrag zum gemeinschaftlichen Erfolg [Mol23, S. 18].

Nachdem die marginalen Beiträge jedes Teilnehmers für die verschiedenen Koalitionen festgestellt wurden, ist der nächste Schritt, die Shapley-Werte zu bestimmen, welche eine faire Aufteilung des Gesamtgewinns erlauben. Hierzu wird jede mögliche Permutation betrachtet, in der die Spieler der Koalition beitreten könnten. Jede dieser Permutationen liefert unterschiedliche marginale Beiträge für die Spieler, je nach der Reihenfolge ihres Beitritts, wie Tabelle 2 zeigt [Mol23, S. 19].

Im Falle des Beispiels mit Anna, Ben und Carla bedeutet dies, dass alle möglichen Reihenfolgen berücksichtigt werden müssen, in denen sie zum ersten Platz beigetragen haben könnten. Die Shapley-Werte werden als Durchschnitt der marginalen Beiträge über alle Permutationen berechnet. Dies gewährleistet, dass jeder Spieler

einen Anteil des Preisgeldes erhält, der seinem durchschnittlichen Beitrag zum Erfolg entspricht [Mol23, S. 20].

Bei drei Teilnehmern exisitieren $3! = 3 \cdot 2 \cdot 1 = 6$ Permutationen:

1. Anna, Ben, Carla
2. Anna, Carla, Ben
3. Ben, Anna, Carla
4. Carla, Anna, Ben
5. Ben, Carla, Anna
6. Carla, Ben, Anna

Jede Permutation entspricht einer Koalitionsbildung. Anna wird in zwei Koalitionsbildungen (1. und 2.) einer leeren Koalition hinzugefügt, da Sie die erste ist, die der Koalition beitritt. In weiteren zwei Koalitionsbildungen (5. und 6.) wird Anna der bestehenden Koalition aus Ben und Carla, respektive Carla und Ben hinzugefügt. In den beiden übrigen Koalitionsbildungen wird Anna einmal der Koaliton bestehend aus Ben (3.) und einmal der Koalition bestehend aus Carla (4.) hinzugefügt.

Zusammen mit Tabelle 2 lässt sich nun der Shapley-Wert mit den gewichteten durchschnittlichen marginalen Beiträge für Anna berechnen:

$$\frac{1}{6}(\underbrace{2 \cdot 500\,€}_{\mathrm{A} \to \{\emptyset\}} + \underbrace{1 \cdot 0\,€}_{\mathrm{A} \to \{B\}} + \underbrace{1 \cdot 750\,€}_{\mathrm{A} \to \{C\}} + \underbrace{2 \cdot 500\,€}_{\mathrm{A} \to \{B,C\}}) \approx 458,33\,€. \tag{2.1}$$

Analog gilt dies für Ben:

$$\frac{1}{6}(\underbrace{2 \cdot 750\,€}_{\mathrm{B} \to \{\emptyset\}} + \underbrace{1 \cdot 250\,€}_{\mathrm{B} \to \{A\}} + \underbrace{1 \cdot 500\,€}_{\mathrm{B} \to \{C\}} + \underbrace{2 \cdot 250\,€}_{\mathrm{B} \to \{A,C\}}) \approx 458,33\,€, \tag{2.2}$$

und Carla:

$$\frac{1}{6}(\underbrace{2 \cdot 0\,€}_{\mathrm{C} \to \{\emptyset\}} + \underbrace{1 \cdot 250\,€}_{\mathrm{C} \to \{A\}} + \underbrace{1 \cdot (-250)\,€}_{\mathrm{C} \to \{B\}} + \underbrace{2 \cdot 250\,€}_{\mathrm{C} \to \{A,B\}}) \approx 83,33\,€. \tag{2.3}$$

Auf Basis der gewichteten durchschnittlichen marginalen Beiträge lässt sich feststellen, dass Anna und Ben jeweils einen Shapley-Wert von ungefähr 458,33 € erhalten, während Carla einen Shapley-Wert von etwa 83,33 € zugewiesen bekommt. Diese Werte spiegeln den fairen Anteil jedes Teilnehmers an der Gesamtprämie wider, basierend auf ihrem individuellen Beitrag zum Erfolg des Teams [Mol23, S. 20]. Mit dieser konkreten Anwendung der Shapley-Werte auf ein alltagsnahes Beispiel wird nun die zugrunde liegende Theorie und die formale Definition der Shapley-Werte, die diese Berechnungen ermöglichen, detaillierter betrachtet.

2.2. Formale Definition

Die Menge $\mathcal{N} = \{1, \ldots, n\}$ repräsentiert eine endliche Anzahl von Spielern, wobei $n := |\mathcal{N}|$ für die Gesamtzahl der Elemente in dieser Menge steht. Die Koalitionsfunktion v ordnet jeder Teilmenge von $\mathcal{N}$ eine reelle Zahl zu, wobei die leere Menge den Wert 0 zugewiesen bekommt:

$$\begin{array}{rcccc} v & : & \mathcal{P}(\mathcal{N}) & \longrightarrow & \mathbb{R} \\ & : & v(\emptyset) & \mapsto & 0. \end{array}$$

Eine nicht leere Teilmenge der Spieler $\mathcal{S} \subseteq \mathcal{N}$ heißt Koalition. $\mathcal{N}$ selbst bezeichnet die große Koalition. Der Term $v(\mathcal{S})$ wird als der Wert der Koalition $\mathcal{S}$ bezeichnet. Der Shapley-Wert weist jedem Mitglied der Spielermenge $\mathcal{N}$ eine spezifische Auszahlung im Rahmen des Spiels v zu.

Der marginale Beitag eines Spielers $i \in \mathcal{N}$, also der Wertbeitrag eines Spielers zu einer Koalition $\mathcal{S} \subseteq \mathcal{N}$, durch seinen Beitritt, ist

$$v(\mathcal{S} \cup \{i\}) - v(\mathcal{S}). \tag{2.4}$$

Sei $i =$ Anna und $\mathcal{S} = \{\text{Ben}\}$, dann ist $v(\{\text{Ben}\} \cup \{\text{Anna}\}) - v(\{\text{Ben}\})$ das zusätzliche Preisgeld, welches gewonnen wird, wenn Anna der Koalition mit Ben beitritt.

Der Wert eines Spielers i nach Shapley wird durch das gewichtete Mittel seiner marginalen Beiträge über alle möglichen Koalitionen bestimmt:

$$\varphi_i(\mathcal{N}, v) = \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{i\}} \underbrace{\frac{|\mathcal{S}|! \cdot (n - 1 - |\mathcal{S}|)!}{n!}}_{\text{Gewicht}} \underbrace{v(\mathcal{S} \cup \{i\}) - v(\mathcal{S})}_{\substack{\text{marginaler Beitrag von} \\ \text{Spieler } i \text{ zur Koalition } \mathcal{S}}} . \tag{2.5}$$

Die Summationsnotation $\sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{i\}}$ erfasst die marginalen Beiträge, die der Spieler i zu allen Koalitionen leistet, welche diesen noch nicht einschließen. Die Verwendung von $\mathcal{N} \setminus \{i\}$ stellt sicher, dass Spieler i nur für jene Koalitionen berücksichtigt wird, zu denen er noch beitragen kann [Mol23, S. 22]. Im Falle von Anna etwa, beziehen sich die Berechnungen auf die Koalitionen bestehend aus der leeren Koalition $\emptyset$, aus {Ben}, {Carla}, oder beiden zusammen {Ben, Carla} (vgl. Berechnung 2.1).

Der Ausdruck $\frac{|\mathcal{S}|! \cdot (n-1-|\mathcal{S}|)!}{n!}$ in der Shapley-Wert-Berechnung reflektiert den Gewichtungsfaktor für die marginalen Beiträge eines Spielers. Hierbei gibt $|\mathcal{S}|!$ die Permutationen der Spieler innerhalb der Koalition $\mathcal{S}$ an, während $(n - 1 - |\mathcal{S}|)!$ die Anordnungen der außenstehenden Spieler repräsentiert, nachdem der betrachtete Spieler beigetreten ist. Der Bruchteil $\frac{1}{n!}$ normalisiert diesen Wert über alle möglichen Koalitionszusammensetzungen, wodurch die Wahrscheinlichkeit der Bildung einer spezifischen Koalition ausgedrückt wird [Mol23, S. 22].

Betrachten wir Anna als den Spieler i und die Koalition $\mathcal{S} = \{\text{Ben, Carla}\}$. Der Ausdruck $\frac{|\mathcal{S}|!\cdot(n-1-|\mathcal{S}|)!}{n!}$ berechnet den Gewichtungsfaktor für Annas marginalen Beitrag zur Koalition $\mathcal{S}$. In diesem Fall ist $|\mathcal{S}| = 2$ und $n = 3$. Somit ergibt sich $|\mathcal{S}|! = 2!$ und $n - 1 - |\mathcal{S}| = 0!$, da nach dem Beitritt von Anna keine weiteren Spieler übrig sind. Der Normalisierungsfaktor ist $n! = 3! = 6$. Daraus folgt:

$$\frac{2! \cdot 0!}{3!} = \frac{2 \cdot 1}{6} = \frac{1}{3}. \tag{2.6}$$

Dies bedeutet, dass unter allen möglichen Permutationen der Spielerreihenfolge, Annas Beitritt zu der Koalition {Ben, Carla} in genau ein Drittel der Fälle geschieht. Somit wird ihr marginaler Beitrag mit diesem Faktor gewichtet, um den Shapley-Wert zu berechnen (vgl. Berechnung 2.1) [Mol23, S. 21f].

2.3. Axiomatische Grundlage

Nachdem die Berechnung des Shapley-Werts für das Beispiel konkretisiert wurde, ist es nun von Bedeutung, die zugrundeliegenden Axiome zu betrachten, welche die theoretische Rechtfertigung für die Methode liefern. Der Shapley-Wert wird nicht nur durch seine Berechnungsmethode, sondern auch durch eine Reihe von Axiomen charakterisiert, die seine Fairness und Kohärenz im Kontext kooperativer Spiele sicherstellen. Lloyd Shapley leitete den Shapley-Wert ursprünglich aus diesen Axiomen ab und bewies, dass dieser der einzige ist, der den Axiomen genügt[3]. Dies Axiome der Effizienz, Symmetrie, Null-Spieler-Eigenschaft und der Additivität sind wesentliche Bestandteile, die die Einzigartigkeit und die wünschenswerten Eigenschaften des Shapley-Werts als Lösungskonzept definieren [Mol23, S. 22].

Effizienz Der Wert der großen Koalition wird an die Spieler verteilt:

$$\sum_{i \in \mathcal{N}} \varphi_i(\mathcal{N}, v) = v(\mathcal{N}). \tag{2.7}$$

Dies bedeutet, dass die Summe der Shapley-Werte aller Spieler dem Gesamtwert entspricht, den die Koalition aller Spieler zusammen erreichen kann. Der Gesamtwert, den die große Koalition $\mathcal{N}$, bestehend aus Anna, Ben und Carla, generiert, wird komplett unter den Spielern aufgeteilt [Mol23, S. 22]. Unter Vernachlässigung minimaler Rundungsdifferenzen entspricht die Summe der Shapley-Werte, berechnet in den Gleichungen 2.1, 2.2 und 2.3, dem kollektiven Ertrag der großen Koalition:

$$458,33€ + 458,33€ + 83,33€ \approx 1000€. \tag{2.8}$$

[3] Eine detaillierte Darstellung dieser Axiome und des Beweises ihrer Einzigartigkeit findet sich in Shapleys Originalarbeit, deren umfassende Behandlung jedoch den Rahmen dieser Arbeit überschreiten würde [Sha53, S. 307-318].

Symmetrie Zwei Spieler i und j, die die gleichen marginalen Beiträgen zu jeder Koalition haben, bekommen den gleichen Wert zugewiesen:

$$v(\mathcal{S} \cup \{i\}) = v(\mathcal{S} \cup \{j\}),\ \forall\, \mathcal{S} \subseteq \mathcal{N} \setminus \{i, j\} \Rightarrow \varphi_i(\mathcal{N}, v) = \varphi_j(\mathcal{N}, v). \tag{2.9}$$

Obwohl Anna und Ben den gleichen Shapley-Wert erhalten, ist dies nicht auf das Symmetrieaxiom zurückzuführen, da ihre marginalen Beiträge zu den Koalitionen variieren. Zum Beispiel leistet Anna keinen Beitrag zur Koalition, wenn Ben bereits Teil davon ist, während Ben einen positiven Beitrag leistet, wenn Anna bereits zur Koalition gehört (vgl. Tabelle 2). Dies zeigt, dass die Gleichheit ihrer Shapley-Werte ein Ergebnis der spezifischen Zahlenkonstellation in diesem Szenario ist und nicht aus der symmetrischen Interaktion zwischen den beiden Spielern resultiert.

Null-Spieler-Eigenschaft (Dummy-Spieler-Eigenschaft) Ein Spieler i, der zu jeder Koalition nichts beiträgt, erhält den Wert Null:

$$v(\mathcal{S} \cup \{i\}) = v(\mathcal{S}),\ \forall\, \mathcal{S} \subseteq \mathcal{N} \setminus \{i\} \Rightarrow \varphi_i(\mathcal{N}, v) = 0. \tag{2.10}$$

Dies stell sicher, dass ein Spieler, der keinen Beitrag leistet, auch nicht belohnt wird.

Additivität Falls sich das Spiel in zwei separate Spiele mit den Koalitionsfunktionen v und w aufspalten lässt, ergibt sich die Auszahlung eines jeden Spielers im kombinierten Spiel aus der Addition seiner Auszahlungen aus den einzelnen Spielen:

$$\varphi_i(\mathcal{N}, v + w) = \varphi_i(\mathcal{N}, v) + \varphi_i(\mathcal{N}, w). \tag{2.11}$$

Wenn Anna, Ben und Carla neben dem ersten Wettbewerb an einem zweiten, unabhängigen Wettbewerb teilnehmen, besagt das Additivitätsaxiom, dass die Shapley-Werte jedes Spielers aus beiden Wettbewerben einfach die Summe ihrer individuellen Shapley-Werte aus jedem einzelnen Wettbewerb sind. Dies impliziert, dass die faire Aufteilung der Gewinne aus beiden Wettbewerben konsistent bleibt, indem die aus dem ersten Wettbewerb abgeleiteten Prinzipien auf den zweiten Wettbewerb übertragen und dann addiert werden [RWB$^+$22, Mol22, S. 5573, S.22f].

3. Von Shapley-Werten zu SHAP: Brückenschlag zur Modellinterpretation

Im Rahmen der kooperativen Spieltheorie ermöglichen die Shapley-Werte eine faire Verteilung des kollektiv erwirtschafteten Nutzens auf die beteiligten Akteure. Diese Methodik findet eine analoge Anwendung in der Welt des maschinellen Lernens, um die Beiträge einzelner Merkmale zur Vorhersageleistung eines Modells zu bewerten [Mol23, S. 26]. Hier wird die Terminologie der Shapley-Werte in den Kontext von Machine Learning Modellen übertragen, wobei jedes Merkmal als „Spieler“ betrachtet wird, dessen Beitrag zur „Auszahlung“ – der Vorhersage des Modells – evaluiert werden soll.

Tabelle 3.: Terminologie der originären Shapley-Werte im Kontext des maschinellen Lernens.

Terminologie Konzept	Terminologie Machine Learning	Ausdruck
Spieler	Merkmal Index	j
Anzahl aller Spieler	Anzahl aller Merkmale	p
Große Koalition	Menge aller Merkmale	$\mathcal{N} = \{1, \ldots, p\}$
Koalition	Menge von Merkmalen	$\mathcal{S} \subseteq \mathcal{N}$
Größe der Koalition	Anzahl der Merkmale in der Koalition $\mathcal{S}$	$\lvert\mathcal{S}\rvert$
Spieler, die nicht in der Koalition sind	Merkmale, die nicht in der Koalition enthalten sind	$C : C = \mathcal{N} \setminus \mathcal{S}$
-	Modell, Vorhersagefunktion	f
Koalitionsfunktion	Vorhersage für Merkmalswerte in der Koalition $\mathcal{S}$ abzüglich der Vorhersage im Mittel	$v_{f,x^{(i)}}(\mathcal{S})$
Auszahlung	Vorhersage für eine Beobachtung $x^{(i)}$ abzüglich der Vorhersage im Mittel	$f(x^{(i)}) - \mathbb{E}(f(X))$
Shapley-Wert	Beitrag des Merkmals j zur Auszahlung des Modells für eine Beobachtung $x^{(i)}$	$\varphi_j^{(i)}(\mathcal{N}, f)$

Quelle: [Mol23, S. 26].

Um diesen Beitrag präzise zu quantifizieren, greift man auf den Prozess der Marginalisierung zurück. Dieser besteht darin, über die Wahrscheinlichkeitsverteilungen der Merkmale zu integrieren, wodurch der Einfluss eines jeden Merkmals auf die Vorhersage isoliert und unabhängig von der spezifischen Zusammensetzung der anderen Merkmale bewertet werden kann. Diese Methodik ermöglicht es nicht nur, diskrete Merkmale, sondern auch solche mit kontinuierlichen Verteilungen zu berücksichtigen. Die Integration über kontinuierliche Verteilungen ist ein fundamentaler Schritt, um den tatsächlichen Beitrag der Merkmale in einer Vielzahl von Anwendungsfällen im

maschinellen Lernen zu erfassen [Mol23, S. 29].

Die Koalitionsfunktion $v_{f,x^{(i)}}(\mathcal{S})$ für ein gegebenes Modell f und eine Beobachtung $x^{(i)}$ ist definiert als:

$$v_{f,x^{(i)}}(\mathcal{S}) = \int_{\mathbb{R}} f(x_{\mathcal{S}}^{(i)} \cup X_C) d\mathbb{P}_{X_C} - \mathbb{E}(f(X)). \tag{3.1}$$

Diese Funktion berechnet den erwarteten Wert der Vorhersage des Modells f, wenn nur eine Teilmenge $\mathcal{S}$ der Merkmale genutzt wird, um die Vorhersage für die spezifische Beobachtung $x^{(i)} \in \mathbb{R}^p$ zu treffen. Das Integral $\int_{\mathbb{R}}$ repräsentiert die Berechnung dieses erwarteten Wertes über alle möglichen Werte der Merkmale, die nicht in $\mathcal{S}$ enthalten sind (X_C), gewichtet durch deren Wahrscheinlichkeitsverteilung $\mathbb{P}_{X_C}$. Die Differenz zum Erwartungswert der Vorhersagen über alle Merkmale $\mathbb{E}(f(X))$ zeigt, wie viel die spezifische Menge an Merkmalen $\mathcal{S}$ zur Vorhersage beiträgt [Mol22, Mol23, S. 221, S. 27].

Der marginale Beitrag eines Merkmals j zu einer Koalition $\mathcal{S}$ ist somit:

$$\begin{aligned} v_{f,x^{(i)}}(\mathcal{S} \cup \{j\}) - v_{f,x^{(i)}}(\mathcal{S}) &= \int_{\mathbb{R}} f(x_{\mathcal{S}\cup\{j\}}^{(i)} \cup X_{C\setminus\{j\}}) d\mathbb{P}_{X_{C\setminus\{j\}}} - \mathbb{E}(f(X)) \\ &\quad - \left(\int_{\mathbb{R}} f(x_{\mathcal{S}}^{(i)} \cup X_C) d\mathbb{P}_{X_C} - \mathbb{E}(f(X)) \right) \\ &= \int_{\mathbb{R}} f(x_{\mathcal{S}\cup\{j\}}^{(i)} \cup X_{C\setminus\{j\}}) d\mathbb{P}_{X_{C\setminus\{j\}}} \\ &\quad - \int_{\mathbb{R}} f(x_{\mathcal{S}}^{(i)} \cup X_C) d\mathbb{P}_{X_C}. \end{aligned} \tag{3.2}$$

Diese Gleichung beschreibt, wie sich der erwartete Wert der Vorhersage ändert, wenn das Merkmal j zu der Menge der Merkmale $\mathcal{S}$ hinzugefügt wird [Mol23, S. 29].

Der Beitrag $\varphi_j^{(i)}(\mathcal{N}, f)$ eines Merkmals j für eine Beobachtung $x^{(i)} \in \mathbb{R}^p$ für die Vorhersage $f(x^{(i)})$ ist gegeben als:

$$\begin{aligned} \varphi_j^{(i)}(\mathcal{N}, f) = &\sum_{\mathcal{S}\subseteq\mathcal{N}\setminus\{j\}} \frac{|\mathcal{S}|! \cdot (p-1-|\mathcal{S}|)!}{p!} \\ &\cdot \left(\int_{\mathbb{R}} f(x_{\mathcal{S}\cup\{j\}}^{(i)} \cup X_{C\setminus\{j\}}) d\mathbb{P}_{X_{C\setminus\{j\}}} - \int_{\mathbb{R}} f(x_{\mathcal{S}}^{(i)} \cup X_C) d\mathbb{P}_{X_C} \right). \end{aligned} \tag{3.3}$$

Diese Formel ist die zentrale Berechnung der SHAP-Werte im maschinellen Lernen [Mol23, S. 29]. Sie summiert den gewichteten, marginalen Beitrag des Merkmals j über alle möglichen Kombinationen der anderen Merkmale. Die Gewichtung berück-

sichtigt die Anzahl der Merkmale in der Koalition $\mathcal{S}$ und die Anzahl der verbleibenden Merkmale, die noch hinzugefügt werden können. Dies ergibt den durchschnittlichen Beitrag des Merkmals j zur Vorhersage für die Beobachtung $x^{(i)}$ [Mol23, S. 30].

Die Integration in der SHAP-Formel ist ein zentraler Schritt, um den erwarteten Beitrag jedes Merkmals unter Berücksichtigung der gesamten Verteilung der Daten zu ermitteln. In diesem Ansatz werden die Merkmale als Zufallsvariablen behandelt. Die Integration erfolgt über die Wahrscheinlichkeitsverteilungen dieser Zufallsvariablen. Durch das Berechnen der erwarteten Vorhersagewerte mit und ohne des jeweiligen Merkmals, unter Einbeziehung der Verteilung aller anderen Merkmale, ermöglicht SHAP eine präzise und umfassende Einschätzung des Einflusses jedes einzelnen Merkmals [Mol23, S. 28].

3.1. Berechnung der SHAP-Werte unter Berücksichtigung der zugrundeliegenden Verteilung

Ein Beispiel soll helfen, die Anwendung von SHAP-Werten im Kontext des maschinellen Lernens zu illustrieren, indem gezeigt wird, wie SHAP-Werte berechnet werden.[4]. Betrachtet wird ein fiktiver Immobilien-Datensatz mit drei Merkmalen: Größe der Immobilie in Quadratmetern (x_1), Anzahl der Zimmer (x_2) und Entfernung zum Stadtzentrum in Kilometern (x_3). Es gibt zwei Beobachtungen in diesem Datensatz:

Tabelle 4.: Merkmale von Beobachtungen in einem Immobilien-Datensatz.

	x_1: Größe (in m^2)	x_2: Anzahl Zimmer	x_3: Entfernung zum Zentrum (in km)
$x^{(1)}$	100	3	5
$x^{(2)}$	150	4	10

Quelle: Eigene Darstellung. Daten sind hypothetisch zu Illustrationszwecken frei erfunden.

Angenommen, das Modell $f(x^{(i)})$ prognostiziert den Preis einer Immobilie in Euro als eine lineare Kombination der Merkmale:

$$f(x^{(i)}) = 5x_1^{(i)} + 20x_2^{(i)} - 2x_3^{(i)}. \tag{3.4}$$

Die Vorhersagen für die beiden Beobachtungen lauten dann:

$$\begin{aligned} f(x^{(1)}) &= 5x_1^{(1)} + 20x_2^{(1)} - 2x_3^{(1)} \\ &= 5 \cdot 100 + 20 \cdot 3 - 2 \cdot 5 \\ &= 550\,€, \end{aligned} \tag{3.5}$$

[4] In Anlehnung an das Beispiel aus Kapitel 8.5.1 „General Idea“ [Mol22, S.215f].

und

$$\begin{aligned} f(x^{(2)}) &= 5x_1^{(2)} + 20x_2^{(2)} - 2x_3^{(2)} \\ &= 5 \cdot 150 + 20 \cdot 4 - 2 \cdot 10 \\ &= 810\,€. \end{aligned} \tag{3.6}$$

Die erwartete Auszahlung des Modells $\mathbb{E}(f(X))$ wird berechnet als:

$$\begin{aligned} \mathbb{E}(f(X)) &= 5 \cdot \mathbb{E}(X_1) + 20 \cdot \mathbb{E}(X_2) - 2 \cdot \mathbb{E}(X_3) \\ &= 5 \cdot 125 + 20 \cdot 3,5 - 2 \cdot 7,5 \\ &= 680\,€, \end{aligned} \tag{3.7}$$

mit

$$\mathbb{E}(X_j) = \frac{1}{n} \sum_{i=1}^{n} x_j^{(i)}. \tag{3.8}$$

Sei $\mathcal{N} = \{1, 2, 3\}$ die Menge aller Merkmale und die Beobachtung $x^{(1)} = [100, 3, 5]$. Der SHAP-Wert für jedes Merkmal $j \in \mathcal{N}$ wird unter Berücksichtigung der Verteilung der Daten und der Formel 3.3 berechnet:

$$\begin{aligned} \varphi_j^{(1)}(\mathcal{N}, f) = \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{j\}} & \frac{|\mathcal{S}|! \cdot (p - 1 - |\mathcal{S}|)!}{p!} \\ & \cdot \left(\int_{\mathbb{R}} f(x^{(1)}_{\mathcal{S} \cup \{j\}} \cup X_{C \setminus \{j\}}) d\mathbb{P}_{X_{C \setminus \{j\}}} - \int_{\mathbb{R}} f(x^{(1)}_{\mathcal{S}} \cup X_C) d\mathbb{P}_{X_C} \right), \end{aligned} \tag{3.9}$$

wobei $p = |\mathcal{N}| = 3$ die Anzahl der Merkmale ist und X_C die Menge der Merkmale außerhalb der Koalition $\mathcal{S}$ darstellt. Die Integrale repräsentieren die erwartete Vorhersage des Modells über die Verteilung der nicht in der Koalition enthaltenen Merkmale.

Der Beitrag durch das Hinzufügen des Merkmals x_1 zur bestehenden Koalition $\mathcal{S} = \{x_2\}$ wird nach Gleichung 3.9 berechnet als:

$$
\begin{aligned}
\varphi_1^{(1)}(\{x_2\}, f) &= \frac{1! \cdot (3-1-1)!}{3!} \\
&\quad \cdot \int f(x_1, x_2, X_3) d\mathbb{P}(X_3) - \int\int f(X_1, x_2, X_3) d\mathbb{P}(X_1, X_3) \\
&= \frac{1}{6}\Big(f(x_1, x_2, \mathbb{E}(X_3)) - f(\mathbb{E}(X_1), x_2, \mathbb{E}(X_3))\Big) \\
&= \frac{1}{6}\Big(f(100, 3, 7.5) - f(125, 3, 7.5)\Big) \\
&= \frac{1}{6}\Big((5 \cdot 100 + 20 \cdot 3 - 2 \cdot 7,5) - (5 \cdot 125 + 20 \cdot 3 - 2 \cdot 7,5)\Big) \\
&= \frac{1}{6}(545 - 670) \\
&= \frac{1}{6}(-125).
\end{aligned} \tag{3.10}
$$

Im Ausdruck $\int f(x_1, x_2, X_3) d\mathbb{P}(X_3)$ wird das Modell f über die Verteilung von X_3 integriert, während x_1 und x_2 konstant gehalten werden. In einem linearen Modell kann diese Integration durch Einsetzen des Erwartungswertes von X_3 vereinfacht werden, da der Beitrag von X_3 zum Gesamtergebnis des Modells linear ist und somit durch den Erwartungswert repräsentiert werden kann. Ähnlich wird im Term $\int f(X_1, x_2, X_3) d\mathbb{P}(X_1, X_3)$ X_1 und X_3 durch ihre Erwartungswerte ersetzt.

Die in Tabelle 5 dargestellten Kombinationen illustrieren die marginalen Beiträge und SHAP-Werte für jedes Merkmal in jeder möglichen Koalition von Merkmalen, bezogen auf die Beobachtung $x^{(1)}$.

Tabelle 5.: Marginalbeiträge der einzelnen Merkmale zu den möglichen Koalitionen für die Beobachtung $x^{(1)}$.

x_j	$\mathcal{S}$	$v_{f,x^{(1)}}(\mathcal{S})$	$v_{f,x^{(1)}}(\mathcal{S} \cup \{j\})$	$v_{f,x^{(1)}}(\mathcal{S} \cup \{j\}) - v_{f,x^{(1)}}(\mathcal{S})$	Gewicht	$\varphi_j^{(1)}(\mathcal{S}, f)$
x_1	$\emptyset$	680 €	555 €	-125 €	$\frac{1}{3}$	-41,67 €
x_1	$\{x_2\}$	670 €	545 €	-125 €	$\frac{1}{6}$	-20.83 €
x_1	$\{x_3\}$	685 €	560 €	-125 €	$\frac{1}{6}$	-20.83 €
x_1	$\{x_2, x_3\}$	675 €	550 €	-125 €	$\frac{1}{3}$	-41,67 €
x_2	$\emptyset$	680 €	670 €	-10 €	$\frac{1}{3}$	-3,33 €
x_2	$\{x_1\}$	555 €	545 €	-10 €	$\frac{1}{6}$	-1,67 €
x_2	$\{x_3\}$	685 €	675 €	-10 €	$\frac{1}{6}$	-1,67 €
x_2	$\{x_1, x_3\}$	560 €	550 €	-10 €	$\frac{1}{3}$	-3,33 €
x_3	$\emptyset$	680 €	685 €	5 €	$\frac{1}{3}$	1,67 €
x_3	$\{x_1\}$	555 €	560 €	5 €	$\frac{1}{6}$	0,83 €
x_3	$\{x_2\}$	670 €	675 €	5 €	$\frac{1}{6}$	0,83 €
x_3	$\{x_1, x_2\}$	545 €	550 €	5 €	$\frac{1}{3}$	1,6 €

Quelle: Eigene Darstellung.

Diese Analyse ist ebenso auf die Beobachtung $x^{(2)}$ anwendbar und erfordert eine analoge Vorgehensweise, wie in Tabelle 6 dargestellt.

Die in den Tabellen 5 und 6 abgebildeten SHAP-Werte für die Beobachtungen $x^{(1)}$ und $x^{(2)}$ zeigen eine interessante Symmetrie. Für jedes Merkmal entspricht der SHAP-Wert für $x^{(2)}$ genau dem negativen Wert für $x^{(1)}$. Die beobachtete Inversion

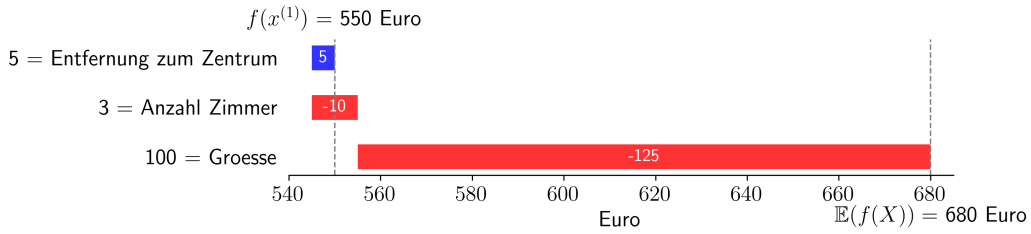
Tabelle 6.: Marginalbeiträge der einzelnen Merkmale zu den möglichen Koalitionen für die Beobachtung $x^{(2)}$.

x_j	$\mathcal{S}$	$v_{f,x^{(2)}}(\mathcal{S})$	$v_{f,x^{(2)}}(\mathcal{S} \cup \{j\})$	$v_{f,x^{(2)}}(\mathcal{S} \cup \{j\}) - v_{f,x^{(2)}}(\mathcal{S})$	Gewicht	$\varphi_j^{(2)}(\mathcal{S}, f)$
x_1	$\emptyset$	680 €	805 €	125 €	$\frac{1}{3}$	41,67 €
x_1	$\{x_2\}$	690 €	815 €	125 €	$\frac{1}{6}$	20.83 €
x_1	$\{x_3\}$	675 €	800 €	125 €	$\frac{1}{6}$	20.83 €
x_1	$\{x_2, x_3\}$	685 €	810 €	125 €	$\frac{1}{3}$	41,67 €
x_2	$\emptyset$	680 €	690 €	10 €	$\frac{1}{3}$	3,33 €
x_2	$\{x_1\}$	805 €	815 €	10 €	$\frac{1}{6}$	1,67 €
x_2	$\{x_3\}$	675 €	685 €	10 €	$\frac{1}{6}$	1,67 €
x_2	$\{x_1, x_3\}$	800 €	810 €	10 €	$\frac{1}{3}$	3,33 €
x_3	$\emptyset$	680 €	675 €	-5 €	$\frac{1}{3}$	-1,67 €
x_3	$\{x_1\}$	805 €	800 €	-5 €	$\frac{1}{6}$	-0,83 €
x_3	$\{x_2\}$	690 €	685 €	-5 €	$\frac{1}{6}$	-0,83 €
x_3	$\{x_1, x_2\}$	815 €	810 €	-5 €	$\frac{1}{3}$	-1,67 €

Quelle: Eigene Darstellung.

der SHAP-Werte zwischen den Beobachtungen $x^{(1)}$ und $x^{(2)}$ lässt sich durch die zentrale Rolle des Erwartungswerts des Modells erklären. Das Modell prognostiziert im Mittel einen Immobilienpreis von 680 €. In einem Szenario, in dem nur zwei Beobachtungen vorhanden sind, spiegeln die Beobachtungen $x^{(1)}$ und $x^{(2)}$ entgegengesetzte Abweichungen vom Erwartungswert wider. Die SHAP-Werte, die den Beitrag jedes Merkmals zur Abweichung der Vorhersage vom Mittelwert messen, zeigen daher für $x^{(1)}$ und $x^{(2)}$ genau entgegengesetzte Werte. Dieses Phänomen resultiert aus der Tatsache, dass die Merkmalsbeiträge in Bezug auf den Erwartungswert berechnet werden und unsere beispielhaften Beobachtungen symmetrisch um diesen Mittelwert verteilt sind. Folglich heben sich ihre Beiträge im Kontext unseres linearen Modells exakt auf, was die Konsistenz und Zuverlässigkeit der SHAP-Wert-Berechnung unterstreicht.

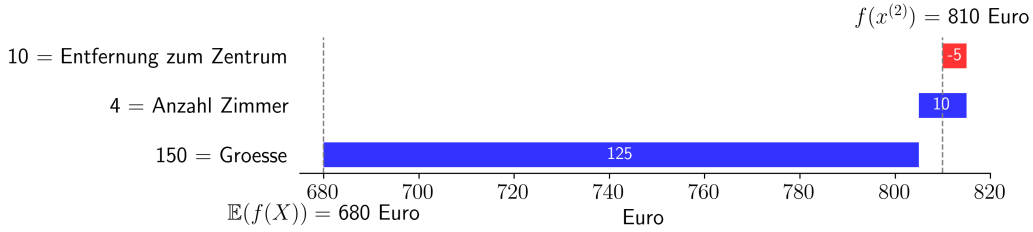
Abbildung 1.: Beitrag der Merkmale $x_{j\in\{1,2,3\}}$ zur Modellvorhersage $f(x^{(1)})$.



Quelle: Eigene Darstellung, A.2.

Das Modell $f(x^{(i)})$ prognostiziert im Mittel einen Immobilienpreis von 680 €. Im Vergleich zur Verteilung des jeweiligen Merkmals, reduziert die Größe der Immobilie ($x_1^{(1)}$) und die Anzahl der Zimmer ($x_2^{(1)}$) die Prognose des Preises für die Immobilie $x^{(1)}$ um insgesamt 135 €, während die Entfernung zum Stadtzentrum ($x_3^{(1)}$) den Preis der Immobilie um 5 € erhöht, wie in Abbildung 1 veranschaulicht. Abbildung 2 visualisiert die SHAP-Werte für die Immobilie $x^{(2)}$:

Abbildung 2.: Beitrag der Merkmale $x_{j\in\{1,2,3\}}$ zur Modellvorhersage $f(x^{(2)})$.



Quelle: Eigene Darstellung, A.2.

3.2. Kontextualisierte axiomatische Grundlage der Shapley-Werte

Die in Tabellen 5 und 6 präsentierten Ergebnisse bieten eine Grundlage, um die Konformität der SHAP-Werte mit den etablierten Axiomen der Shapley-Werte, wie sie im Kapitel 2.3 erläutert wurden, zu beurteilen. Die Axiome der SHAP-Werte stellen eine adaptierte und kontextualisierte Anwendung dieser Prinzipien auf die Interpretation von Modellvorhersagen dar [AFSS19, S. 16f].

Effizienz Das Effizienzaxiom besagt, dass die Summe der SHAP-Werte aller Merkmale für eine gegebene Beobachtung $x^{(i)}$ gleich der Differenz zwischen der Modellvorhersage für diese Beobachtung $f(x^{(i)})$ und der durchschnittlichen Modellvorhersage $\mathbb{E}(f(X))$ sein muss:

$$\sum_{j=1}^{p} \varphi_j^{(i)}(\mathcal{N}, f) = f(x^{(i)}) - \mathbb{E}(f(X)), \tag{3.11}$$

[Mol22, S. 221]. Für die Beobachtung $x^{(1)}$ aus Kapitel 3.1 und den Berechnungen für $f(x^{(1)})$ (Gleichung 3.5), sowie $\mathbb{E}(f(X))$ (Gleichung 3.7) ergibt sich:

$$\begin{aligned} \sum_{j=1}^{3} \varphi_j^{(1)}(\mathcal{N}, f) &= -130\,€ \\ f(x^{(1)}) - \mathbb{E}(f(X)) &= 550\,€ - 680\,€ = -130\,€, \end{aligned} \tag{3.12}$$

womit das Effizienzaxiom erfüllt ist. Die Differenz der Vohersage einer konkreten Beobachtung zur durchschnittlichen Modellvorhersage wird auf alle Merkmale verteilt.

Symmetrie Das Symmetrieaxiom fordert, dass zwei Merkmale i und j, die in jeder Koalition denselben Beitrag leisten, auch denselben SHAP-Wert erhalten müssen.

In dem hier betrachteten Fall der Immobilienpreisprognose würde dies bedeuten, dass zwei Merkmale, beispielsweise die Größe einer Immobilie und die Anzahl der Zimmer, die immer den gleichen Einfluss auf den Preis hätten, unabhängig von der Kombination anderer Merkmale, auch identische SHAP-Werte haben müssen:

$$v(\mathcal{S} \cup \{i\}) = v(\mathcal{S} \cup \{j\}),\ \forall \mathcal{S} \subseteq \mathcal{N} \setminus \{i, j\} \Rightarrow \varphi_i(\mathcal{N}, v) = \varphi_j(\mathcal{N}, v), \tag{3.13}$$

[Mol22, S. 221]. Dies wird durch die Tabellen 5 und 6 nicht illustriert, da jedes Merkmal einen unterschiedlichen Beitrag liefert, was die Anwendung dieses Axioms in diesem speziellen Fall ausschließt.

Null-Spieler-Eigenschaft (Dummy-Spieler-Eigenschaft) Ein Merkmal i, das keinen Einfluss auf die Modellvorhersage hat, erhält gemäß der Null-Spieler-Eigenschaft einen SHAP-Wert von null. Im Kontext des Beispiels würde ein Merkmal, das keine Veränderung in der Vorhersage bewirkt, unabhängig von den anderen Merkmalen, einen SHAP-Wert von null erhalten:

$$v(\mathcal{S} \cup \{i\}) = v(\mathcal{S}),\ \forall \mathcal{S} \subseteq \mathcal{N} \setminus \{i\} \Rightarrow \varphi_i(\mathcal{N}, v) = 0, \tag{3.14}$$

[Mol22, S. 222]. In der fiktiven Datenlage der Tabellen 5 und 6 hat jedes Merkmal einen gewissen Einfluss, sodass die Null-Spieler-Eigenschaft hier nicht beobachtet werden kann.

Additivität Das Additivitätsaxiom ist ein zentrales Konzept, das die Konsistenz von Shapley-Werten über die Zusammensetzung von Spielen hinweg beschreibt. Es garantiert, dass für zwei separate Spiele oder Modelle v und w, die Summe der SHAP-Werte eines Merkmals über beide Spiele seinem SHAP-Wert im kombinierten Spiel entspricht:

$$\varphi_i(\mathcal{N}, v + w) = \varphi_i(\mathcal{N}, v) + \varphi_i(\mathcal{N}, w). \tag{3.15}$$

In Machine Learning-Modellen wie dem Random Forest, besonders bei Ensemble-Modellen, ist die Additivität relevant. Ein Random Forest besteht aus unabhängigen Entscheidungsbäumen, die zusammenarbeiten, um Vorhersagen zu treffen [Mol23, S. 97]. Die SHAP-Werte der einzelnen Bäume können als additive Beiträge betrachtet werden, die den Gesamteinfluss eines Merkmals auf die Modellvorhersage zusammenfassen [Mol23, S. 32].

4. Komplexitätsbewältigung bei der Berechnung von SHAP-Werten

Die direkte Berechnung von SHAP-Werten kann bei Modellen mit einer großen Anzahl an Merkmalen rechnerisch anspruchsvoll sein. Die Herausforderung bei der Berechnung von SHAP-Werten in realen Szenarien erwächst aus zwei zentralen Problembereichen: der Anzahl der Merkmale und der Notwendigkeit, über diese Merkmale zu integrieren.

Die Anzahl möglicher Koalitionen von Merkmalen wächst exponentiell mit 2^p an, was bei einer großen Anzahl von Merkmalen p schnell zu einer nicht handhabbaren Berechnungskomplexität führt. Hinzu kommt die Schwierigkeit, dass für eine exakte Berechnung der SHAP-Werte eine Integration über die Merkmalsverteilungen erforderlich ist, was jedoch voraussetzt, dass die Verteilung der Merkmale bekannt ist. In der Praxis verfügt man meist nur über eine Stichprobe der Daten, ohne genaue Kenntnis der zugrundeliegenden Verteilungen [Mol23, S. 33].

4.1. Linearer SHAP Estimator für lineare Modelle

Im Kontext linearer Modelle, wie dem in dieser Arbeit verwendeten Immobilienpreis-Beispiel, ist die vollständige Ermittlung aller möglichen Kombinationen von Merkmalen und deren Beiträgen jedoch nicht notwendig.

Anstelle der komplexen Integration über die Verteilungen aller Merkmale, kann der Fokus auf die Unterschiede in den Modellvorhersagen gelegt werden, die sich aus dem Hinzufügen oder Entfernen einzelner Merkmale ergeben. Hierbei werden anstelle der spezifischen Werte der nicht in der betrachteten Koalition enthaltenen Merkmale, die Erwartungswerte herangezogen. Diese Vereinfachung ermöglicht es, den Einfluss jedes Merkmals auf die Modellvorhersage auf eine direktere und rechnerisch weniger aufwendige Weise zu erfassen. Diese Vereinfachung ist für lineare Modelle angemessen, da die Auswirkungen jedes Merkmals auf die Vorhersage des Modells additiv und unabhängig sind [Mol23, S. 48].

Theoretisch könnte daher für jedes Merkmal nur eine Koalition berechnet werden, um den SHAP-Wert zu bestimmen [Mol23, S. 38]. Dies bestätigen die Tabellen 5 und 6 und die daraus ersichtlichen marginalen Beiträge, die über die jeweiligen Merkmale unabhängig der ihnen beigetretenen Koalition konstant sind. So ist der marginale Beitrag für Merkmal $x_1^{(1)}$ zu allen möglichen Koalitionen stets -125.

Ein lineares Modell wird typischerweise in der Form $f(x) = \sum_{j=1}^{p} \beta_j x_j + b$ ausgedrückt, wobei β_j der Gewichtungskoeffizient für das Merkmal j ist und b den Achsenabschnitt (Bias) darstellt. Der lineare SHAP Estimator nutzt diese Koeffizienten, um die Beiträge jedes Merkmals zur Modellvorhersage zu bestimmen. Der SHAP-Wert für ein Merkmal j wird dann definiert als:

$$\varphi_j^{(i)}(f, x) = \beta_j(x_j^{(i)} - \mathbb{E}[X_j]), \tag{4.1}$$

wobei $\mathbb{E}[X_j]$ der Erwartungswert des Merkmals j über den Datensatz ist. Der SHAP-Wert für den Achsenabschnitt (Bias) ist gleich dem Achsenabschnitt des Modells: $\varphi_0(f, x) = b$ [LL17, S. 6].

Aufgrund der Unabhängigkeit der Eingabemerkmale, kann das arithmetische Mittel aus Kapitel 3.1 Gleichung 3.8 als erwartungstreuer Schätzer für den Erwartungswert verwendet werden:

$$\varphi_j^{(i)}(f, x) = \beta_j\Big(x_j^{(i)} - \frac{1}{n}\sum_{k=1}^{n}(x_j^{(k)})\Big), \tag{4.2}$$

Das Effizienzaxiom aus Gleichung 3.11 ist erfüllt:

$$\begin{aligned}\sum_{j=1}^{p} \varphi_j^{(i)}(f, x) &= \sum_{j=1}^{p} \Big(\beta_j x_j^{(i)} - \mathbb{E}[\beta_j X_j]\Big) \\ &= \beta_0 + \sum_{j=1}^{p} \beta_j x_j^{(i)} - (\beta_0 + \sum_{j=1}^{p} \mathbb{E}[\beta_j X_j]) \\ &= f(x) - \mathbb{E}[f(X)],\end{aligned} \tag{4.3}$$

[Mol23, S. 48].

5. Praktische Anwendung von SHAP auf lineare Modelle

In diesem Kapitel wird der Einsatz des SHAP-Frameworks zur Interpretation linearer Modelle im Kontext des maschinellen Lernens untersucht. Lineare Modelle, gekennzeichnet durch ihre Transparenz und einfache Struktur, bilden oft die Basis für das Verständnis komplexerer Algorithmen [Mol22, S. 37]. Dennoch bleibt die Herausforderung bestehen, die Beiträge individueller Merkmale zur Modellvorhersage zu quantifizieren und zu interpretieren.

Die Anwendung von SHAP-Werten ermöglicht es, diesen Herausforderungen zu begegnen und Einblicke in die Modellvorhersagen zu gewähren, die über traditionelle Methoden hinausgehen [Mol22, S. 224]. Dieses Kapitel führt in die Grundlagen des shap-Pakets ein, demonstriert dessen Anwendung auf einen spezifischen Datensatz und diskutiert die Berechnung sowie Interpretation der resultierenden SHAP-Werte.

5.1. Lineare Modelle als analytische Grundlage

In linearen Regressionsmodellen wird die Zielgröße als eine gewichtete Kombination der Eingangsmerkmale bestimmt. Die einfache lineare Struktur dieser Modelle erleichtert das Verständnis der Beziehungen zwischen den Eingangsdaten und den Vorhersagen.

Lineare Modelle sind ein grundlegendes Werkzeug in der statistischen Modellierung und dienen dazu, das Verhältnis zwischen einer abhängigen Variablen, die üblicherweise mit $y^{(i)}$ bezeichnet wird, und einem oder mehreren Prädiktoren, den unabhängigen Variablen x_i, zu erfassen. Diese Beziehungen werden mittels linearer Gleichungen dargestellt, die für jede einzelne Beobachtung i im Datensatz folgendermaßen formuliert werden können:

$$y^{(i)} = \beta_0 + \sum_{j=1}^{p} \beta_j x_j^{(i)} + \epsilon^{(i)}, \tag{5.1}$$

wobei das Ergebnis, das von einem linearen Modell für eine gegebene Beobachtung vorhergesagt wird, sich als Summe der mit Gewichten β_j versehenen Merkmale p ergibt.

Hierbei stellt $y^{(i)}$ den beobachteten Wert der abhängigen Variablen für die Beobachtungseinheit i dar. Der Term β_0 ist der Achsenabschnitt oder y-Achsenabschnitt des Modells, welcher den erwarteten Wert von y darstellt, wenn alle unabhängigen

Variablen x null sind. Die Summe $\sum_{j=1}^{p} \beta_j x_j^{(i)}$ berechnet sich aus den Produkten der Koeffizienten β_j und den Werten der unabhängigen Variablen $x_j^{(i)}$ für jede Beobachtungseinheit i und jeden Prädiktor j, wobei die Koeffizienten β_j den geschätzten Einfluss der entsprechenden unabhängigen Variablen auf die abhängige Variable beschreiben.

Der Fehlerterm $\epsilon^{(i)}$ steht für die Residuen, also die Differenzen zwischen den beobachteten und durch das Modell geschätzten Werten von $y^{(i)}$. Es wird angenommen, dass diese Fehler normalverteilt sind, was bedeutet, dass Abweichungen in beiden Richtungen um den Mittelwert (hier null) mit abnehmender Wahrscheinlichkeit für größere Fehler auftreten [Mol22, S. 37].

In einem linearen Modell stellt der Achsenabschnitt die Basislinie dar, an der die Auswirkungen aller anderen Merkmale gemessen werden. Dieser Wert gibt an, was das Modell für die Zielvariable vorhersagen würde, wenn alle anderen Merkmale nicht vorhanden wären – der Ausgangspunkt der Vorhersage für einen Datensatz, in dem alle anderen Variablen auf null gesetzt sind. Es ist wichtig zu erwähnen, dass der Achsenabschnitt für sich genommen nicht immer eine praktische Bedeutung hat, da es selten vorkommt, dass alle Variablen tatsächlich den Wert null annehmen. Die wahre Aussagekraft des Achsenabschnitts tritt zutage, wenn die Daten so standardisiert wurden, dass ihre Mittelwerte bei null und die Standardabweichung bei eins liegen. Unter diesen Umständen repräsentiert der Achsenabschnitt die erwartete Zielvariable für einen hypothetischen Fall, in dem alle Merkmale ihren Durchschnittswert aufweisen [Mol22, S. 39].

Bei der Betrachtung einzelner Merkmale innerhalb des Modells sagt das Gewicht β_j eines Merkmals, um wie viel sich die Zielvariable $y^{(i)}$ ändert, wenn das Merkmal $x_j^{(i)}$ um eine Einheit erhöht wird – und zwar unter der Annahme, dass alle anderen Merkmale unverändert bleiben. Dies ermöglicht es, den isolierten Effekt eines jeden Merkmals auf die Vorhersage zu verstehen [Mol22, S. 39].

Die optimalen Gewichte, oder Koeffizienten, eines linearen Regressionsmodells werden üblicherweise durch ein Verfahren bestimmt, das als Methode der kleinsten Quadrate (engl. *Ordinary Least Squares*, OLS) bekannt ist [Mol22, S. 37]. Diese Methode sucht die Koeffizienten $\beta_0, \ldots, \beta_p$, welche die Summe der quadrierten Differenzen zwischen den beobachteten Werten der Zielvariablen $y^{(i)}$ und den von dem Modell vorhergesagten Werten minimieren:

$$\hat{\beta} = \arg\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y^{(i)} - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j^{(i)} \right) \right)^2 . \tag{5.2}$$

Das Ergebnis der Minimierung, $\hat{\beta}$, stellt den Vektor der geschätzten Koeffizienten

dar [Mol22, S. 37]. In der vorliegenden Arbeit wird das Python-Paket scikit-learn[5] verwendet, um die lineare Regression durchzuführen und die Koeffizienten $\hat{\beta}$ zu bestimmen.

5.2. Einführung in das shap Python-Paket

Das Python-Paket shap[6] ist eine Open-Source-Bibliothek, die es Nutzern ermöglicht, die Auswirkungen von Merkmalen auf Vorhersagen von maschinellen Lernmodellen zu interpretieren und zu visualisieren. Entwickelt wurde die Bibliothek ursprünglich von Scott Lundberg und weiteren Mitwirkenden im Rahmen der Forschungsarbeit an der University of Washington [LL17]. Das Paket basiert auf dem Konzept der Shapley-Werte aus der kooperativen Spieltheorie und überträgt diese auf den Kontext des maschinellen Lernens, um als Tool für die Interpretierbarkeit und Erklärbarkeit von Modellvorhersagen zu dienen.

Die Kernfunktion des shap-Pakets ist die Berechnung von SHAP-Werten, welche die Auswirkung der Einzelmerkmale auf die Modellvorhersage quantifizieren. Jeder SHAP-Wert ist ein Maß dafür, wie viel jedes Merkmal zur Vorhersage beigetragen hat, im Vergleich zu einer durchschnittlichen Vorhersage über den gesamten Datensatz. Diese Werte sind besonders wertvoll, da sie ein Maß für die Bedeutung jedes Merkmals bieten. Diese Bedeutung kann sowohl lokal für einzelne Vorhersagen als auch global über das gesamte Modell interpretiert werden

Mit shap können Benutzer die Vorhersagen einer Vielzahl von Modellen interpretieren, von linearen Modellen bis hin zu komplexen Konstrukten wie tiefe neuronale Netzwerke. Die Bibliothek bietet eine vielseitige Auswahl an Visualisierungsoptionen, darunter Beeswarm-Plots, Dependence-Plots und Bar-Plots, die es ermöglichen, die SHAP-Werte intuitiv zu verstehen. Eine Übersicht aller Visualisierungsoptionen ist in der Dokumentation des Pakets zu finden[7]. Diese Visualisierungen erleichtern es, Muster und Beiträge einzelner Merkmale zu erkennen, was nicht nur wertvolle Einblicke in die Leistung des Modells bietet, sondern auch zu faireren und transparenteren Modellentscheidungen führen kann.

Im Kapitel 4.1 wurde bereits der LinearExplainer aus dem shap-Paket vorgestellt, ein Beispiel für die verschiedenen Estimators, die das Paket in Form von Explainern bereitstellt. Das Paket bietet eine Vielzahl von Explainern, die auf unterschiedliche Modelltypen zugeschnitten sind. Einer der bemerkenswerten Aspekte von shap ist der auto Modus des Estimators, der automatisch den am besten geeigneten Explainer für das gegebene Modell auswählt. Diese Funktion ist besonders nützlich, da sie die Komplexität der Auswahl des richtigen Explainers reduziert und den Anwendungsprozess vereinfacht. Speziell für Modelle mit ca. 15 Merkmalen[8] wählt der auto

[5] https://scikit-learn.org
[6] https://shap.readthedocs.io
[7] https://shap.readthedocs.io/en/latest/api.html#plots
[8] https://github.com/shap/shap/blob/master/shap/explainers/_exact.py

Modus den exakten Explainer, der präzise SHAP-Werte auf Grundlage aller Daten und Koalitionen berechnet, was für Modelle mit einer geringeren Anzahl von Merkmalen effizient und praktikabel ist [Mol23, S. 40f]. Eine Übersicht der zur Verfügung stehenden shap-Explainern ist in der Dokumentation des Pakets zu finden[9].

5.3. Einführung in den Datensatz

Der Concrete Compressive Strength Datensatz ist eine umfassende Sammlung von Daten, die die Druckfestigkeit von Beton in Bezug auf verschiedene Bestandteile und das Alter des Betons untersucht. Die Bestandteile umfassen Zement, Hochofenschlacke, Flugasche, Wasser, Superplastifikator, groben Zuschlag und feinen Zuschlag [Yeh07].

Ziel der Regression: Das Ziel ist es, die Druckfestigkeit von Beton in Megapascal basierend auf seiner Zusammensetzung und dem Alter vorherzusagen. Diese Vorhersage ist entscheidend, da Beton das Fundament der modernen Welt bildet und seine Festigkeit für die Sicherheit und Langlebigkeit von Bauwerken und damit für den Schutz von Menschenleben von größter Bedeutung ist. Beton spielt eine zentrale Rolle im Bauwesen. Als eines der am meisten verwendeten Materialien weltweit, ist die genaue Vorhersage seiner Festigkeit von entscheidender Bedeutung für die Planung und den Bau sicherer und langlebiger Strukturen [NK21, S. 1].

Der Datensatz umfasst folgende Variablen:

- **cement (Zement)**: Menge an Zement (kg in einem m^3 Gemisch).
- **blast (Hochofenschlacke)**: Menge an Hochofenschlacke (kg in einem m^3 Gemisch).
- **ash (Flugasche)**: Menge an Flugasche (kg in einem m^3 Gemisch).
- **water (Wasser)**: Menge an Wasser (kg in einem m^3 Gemisch).
- **superplasticizer (Superplastifikator)**: Menge an Superplastifikator (kg in einem m^3 Gemisch).
- **coarse (Grober Zuschlag)**: Menge an grobem Zuschlag (kg in einem m^3 Gemisch).
- **fine (Feiner Zuschlag)**: Menge an feinem Zuschlag (kg in einem m^3 Gemisch).
- **age (Alter)**: Alter des Betons in Tagen.
- **strength (Druckfestigkeit)**: Die tatsächliche Druckfestigkeit von Beton (MPa). Dies ist die Zielvariable, die modelliert wird.

[9] https://shap.readthedocs.io/en/latest/api.html#explainers

5.4. Explorative Datenanalyse & Datenaufbereitung

Der Datensatz wurde in Python mithilfe der Bibliothek pandas als Dataframe eingelesen. Der vollständige Quellcode für das Einlesen der Daten sowie alle weiteren Analyseschritte ist im Anhang A.3 dieser Arbeit zu finden.

Tabelle 7 zeigt die ersten fünf Beobachtungen des Datensatzes:

Tabelle 7.: Auszug aus dem Concrete Compressive Strength Datensatz.

Index	cement	blast	ash	water	superplasticizer	coarse	fine	age	strength
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.986111
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.887366
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.269535
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.052780
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.296075

Quelle: Eigene Darstellung, A.3.

Tabelle 8.: Statistische Übersicht des Concrete Compressive Strength Datensatzes.

Variable	mean	std	min	25%	50%	75%	max
cement	278.629	104.345	102.0	190.68	265.0	349.0	540.0
blast	72.043	86.171	0.0	0.0	20.0	142.5	359.4
ash	55.535	64.207	0.0	0.0	0.0	118.27	200.1
water	182.074	21.341	121.75	166.61	185.7	192.94	247.0
superplasticizer	6.032	5.92	0.0	0.0	6.1	10.0	32.2
coarse	974.376	77.58	801.0	932.0	968.0	1031.0	1145.0
fine	772.687	80.34	594.0	724.3	780.0	822.2	992.6
age	45.857	63.735	1.0	7.0	28.0	56.0	365.0
strength	35.250	16.285	2.332	23.524	33.798	44.868	82.599

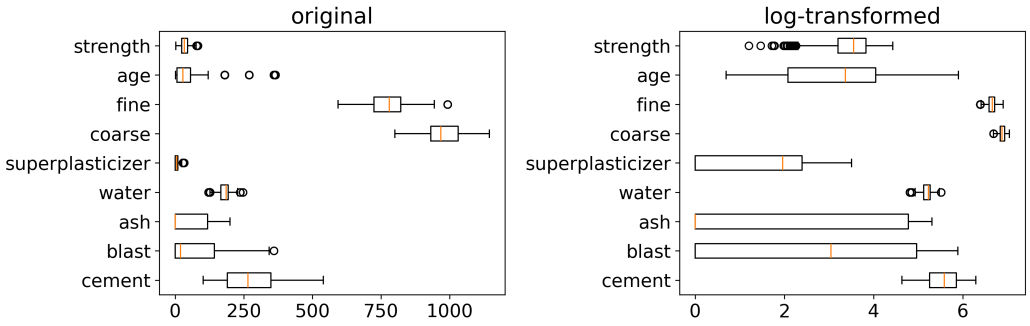
Quelle: Eigene Darstellung, A.3.

Tabelle 8 offenbart eine signifikante Variabilität und Bandbreite in den Werten aller Variablen. Um eine gleichmäßige Verteilung zu erreichen und den Einfluss von Ausreißern zu verringern, werden die Daten einer logarithmischen Transformation unterzogen. Diese Methode dient dazu, die Skalierung der Variablen zu vereinheitlichen und ihre Verteilung zu normalsieren, was die Genauigkeit und Stabilität des Regressionsmodells verbessert [FWL$^+$14, S. 105f]. Abbildung 3 zeigt diese Normalisierung in Form eines Boxplots der jeweiligen Merkmale vor und nach der logarithmischen Transformation.

Abbildung 4 zeigt die Häufigkeitsverteilung aller Merkmale des Datensatzes nach der logarithmischen Transformation.

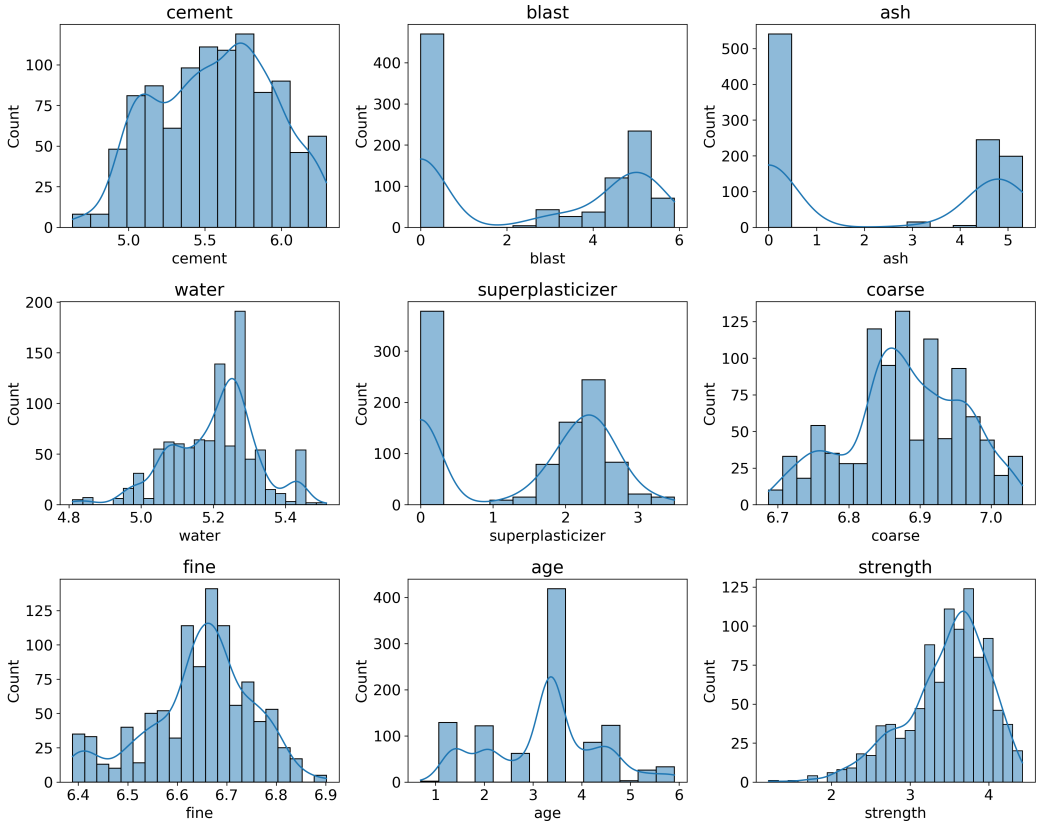
Die Korrelationsmatrix, dargestellt in Abbildung 5, quantifziert die Beziehungen zwischen den einzelnen Bestandteilen des Betons und der Zielvariablen der Betondruckfestigkeit. Die Koeffizienten bewegen sich zwischen -0.61 und 0.63, was auf unter-

Abbildung 3.: Boxplot der Merkmale vor und nach der Log-Transformation.



Quelle: Eigene Darstellung, A.3.

Abbildung 4.: Verteilungen der Merkmale.

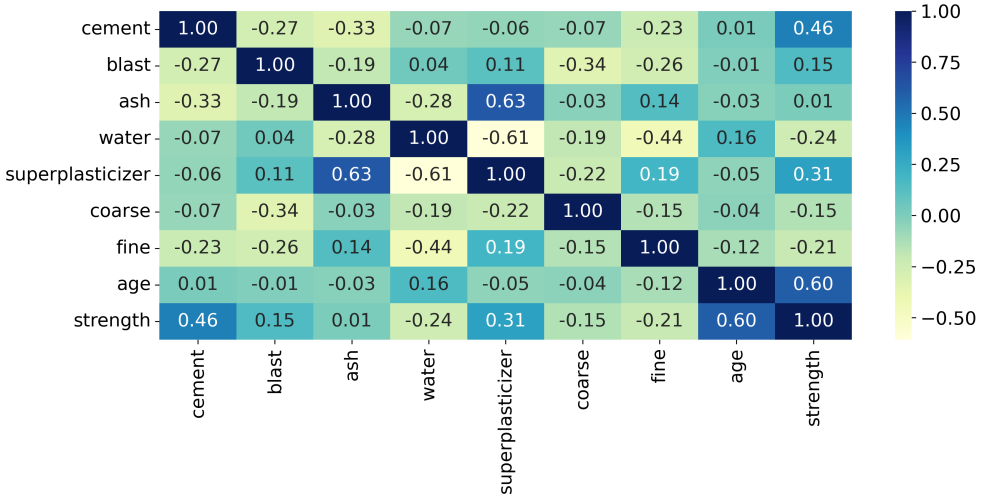


Quelle: Eigene Darstellung, A.3.

schiedlich starke Korrelationen hinweist. Ein markantes Beispiel ist der positive Wert von 0.63 zwischen Asche und Wasser, was eine starke direkte Beziehung nahelegt, während der Wert von -0.61 zwischen Wasser und Superplastifikator auf eine starke umgekehrte Beziehung hindeutet. Die Zielvariable der Betondruckfestigkeit zeigt die stärkste direkte Korrelation mit dem Zementgehalt (0.46), was darauf schließen lässt, dass ein höherer Zementanteil tendenziell zu einer stärkeren Betonmischung führt. Diese Korrelationsmuster sind essentiell für die Modellentwicklung, da sie Aufschluss darüber geben, welche Mischungsbestandteile einen bedeutenden Einfluss auf die

Betondruckfestigkeit haben könnten.

Abbildung 5.: Korrelationsmatrix der Merkmale im Datensatz.



Quelle: Eigene Darstellung, A.3.

5.5. Modellierung der linearen Regression

Um die Beziehung zwischen den unabhängigen Variablen und der Zielvariablen zu untersuchen, wurde ein lineares Regressionsmodell aufgestellt. Zur Bewertung der Vorhersageleistung des Modells und zur Vermeidung von Überanpassung wurde der Datensatz in zwei Teile aufgeteilt: 80% der Daten dienten als Trainingsset zur Anpassung des Modells, während die restlichen 20% als Testset verwendet wurden, um die Modellleistung anhand neuer, unbekannter Daten zu evaluieren [Mol23, S. 51]. Diese Aufteilung erfolgte zufällig, aber reproduzierbar, durch Festlegen eines Seed-Werts für den Zufallszahlengenerator, der eine konsistente Teilung des Datensatzes ermöglicht.

Das Trainingsset wurde dazu verwendet, die Koeffizienten der linearen Regression zu schätzen, die den Einfluss jeder unabhängigen Variablen auf die Zielvariable quantifizieren. Anschließend wurde das Modell mit dem Testset geprüft, um seine Vorhersagegenauigkeit zu bewerten. Die Leistung des Modells wurde anhand von Metriken wie dem mittleren quadratischen Fehler (Mean Squared Error, MSE) gemessen, die ein Maß für die Abweichung der Modellvorhersagen von den tatsächlichen Werten darstellen [Mol22, S. 40].

Codeauschnitt 5.1 und 5.2 zeigen das Trainieren und Testen der zugrundeliegenden Daten eines linearen Regressionsmodells:

Codeauschnitt 5.1: Initialisierung eines linearen Regressionsmodells, A.3.

```
1 def model(X: pd.DataFrame, y: pd.Series) -> (LinearRegression, pd.DataFrame):
2     """
3     Fits a Linear Regression model to the given data.
```

```
    Args:
        X (pd.DataFrame): The feature matrix.
        y (pd.Series): The target variable.

    Returns:
        LinearRegression: The fitted Linear Regression model.
        DataFrame: The coefficients as DataFrame.
    """
    model = LinearRegression()
    model.fit(X, y)

    cdf = pd.DataFrame(model.coef_.round(5), X.columns, columns=['Coefficients'])
    cdf.loc['Intercept'] = model.intercept_.round(5)

    return model, cdf
```

Codeauschnitt 5.2: Training und Testen eines linearen Regressionsmodells, A.3.

```
X = df.drop(['strength'], axis=1)
y = df['strength']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
    random_state=42)

linreg, coef = model(X=X_train, y=y_train)
y_pred = linreg.predict(X_test)
```

5.6. Berechnung von SHAP-Werten

Um SHAP-Werte zu berechnen, wird zunächst ein SHAP-Explainer-Objekt erstellt. In diesem Fall wird der Explainer von SHAP mit dem trainierten linearen Regressionsmodell und dem Trainingsdatensatz initialisiert. Anschließend werden die SHAP-Werte für die Testdaten berechnet, um die Beiträge der einzelnen Merkmale zu analysieren. Der Typ des Explainers wird durch die Art des übergebenen Modells bestimmt. Da in diesem Beispiel ein lineares Modell verwendet wird, wird automatisch ein geeigneter Explainer für lineare Modelle ausgewählt.

Der folgende Codeausschnitt 5.3 zeigt die Initialisierung des SHAP-Explainers und die Berechnung der SHAP-Werte:

Codeauschnitt 5.3: Berechnung von SHAP-Werten für das lineare Regressionsmodell, A.3.

```
explainer = shap.Explainer(linreg.predict, X)
shap_values = explainer(X_test)
```

Das Explainer-Objekt enthält neben den SHAP-Werten (.values), die die Einflüsse der einzelnen Merkmale der Testmenge auf die Modellvorhersage quantifizieren, auch die Basiswerte (.base_values), die die durchschnittliche Vorhersage des Modells darstellen, und die ursprünglichen Merkmalsausprägungen (.data), die für die Berechnung dieser Werte verwendet wurden [Mol23, S. 51].

Dies bildet die Grundlage für den nächsten entscheidenden Schritt: die Visualisierung und tiefere Analyse dieser Werte. Die SHAP-Bibliothek bietet eine Reihe von leistungsstarken Visualisierungswerkzeugen, die es ermöglichen, die Auswirkungen der einzelnen Merkmale auf die Modellvorhersagen intuitiv und verständlich abzubilden.

Im folgenden Kapitel 6 werden diese Visualisierungen im Detail vorgestellt. Anhand von Beeswarm-Plots, Dependence-Plots und Bar-Plots werden die Ergebnisse der SHAP-Analyse dargestellt, die ein umfassendes Bild der Einflüsse und Wichtigkeiten der verschiedenen Merkmale im Kontext des linearen Regressionsmodells bieten.

Die Grafiken wurden mithilfe der shap-Bibliothek wie folgt erzeugt:

Codeauschnitt 5.4: Erzeugen der SHAP Plots, A.3.

```
def plot_shap(shap_values: shap.Explanation, model: LinearRegression, X: pd.
    DataFrame, idx: int) -> None:
    """
    Creates and saves SHAP beeswarm, bar, and waterfall plots.

    Args:
        shap_values (shap.Explanation): SHAP values.
        model (LinearRegression): The Linear Regression model.
        X (pd.DataFrame): The background data for partial dependence plot.
        idx (int): Index for the SHAP waterfall plot.
    """
    plt.figure(figsize=(12, 6))
    shap.plots.beeswarm(shap_values)
    plt.tight_layout()
    plt.savefig('images/shap_beeswarm_plot.png', dpi=300)

    plt.figure(figsize=(12, 6))
    shap.plots.bar(shap_values)
    plt.tight_layout()
    plt.savefig('images/shap_bar_plot.png', dpi=300)

    plt.figure(figsize=(12, 6))
    shap.plots.waterfall(shap_values[idx])
    plt.tight_layout()
    plt.savefig('images/shap_waterfall_plot.png', dpi=300)

    plt.figure(figsize=(12, 6))
    shap.plots.partial_dependence("cement", model.predict, X,
                                  model_expected_value=True,
                                  feature_expected_value=True,
                                  ice=False,
                                  shap_values=shap_values[idx:idx+1,:])
    plt.tight_layout()
    plt.savefig('images/shap_dependence_plot.png', dpi=300)

    plt.figure(figsize=(12, 6))
    shap.plots.scatter(shap_values[:, "cement"])
    plt.tight_layout()
    plt.savefig('images/shap_scatter_plot.png', dpi=300)
```

6. Ergebnisse

In diesem Kapitel werden die Resultate der angewandten linearen Regressionsanalyse zur Vorhersage der Druckfestigkeit von Beton dargestellt. Die Analyse berücksichtigt sowohl die geschätzten Koeffizienten des linearen Modells als auch verschiedene Evaluierungsmetriken wie den mittleren absoluten Fehler (engl. *Mean Absolut Error*, MAE), den mittleren quadratischen Fehler (engl. *Mean Squared Error*, MSE), sowie den sowie die Bestimmtheitsmaße (R^2) für Trainings- und Testdaten. Diese Metriken liefern Aufschluss über die Güte des Modells und die Präzision der Vorhersagen. Zunächst werden die Koeffizienten des linearen Modells interpretiert. Anschließend wird gezeigt, wie SHAP-Werte eine tiefere und detailliertere Analyse ermöglichen.

6.1. Lineares Regressionsmodell

Die Koeffizienten eines linearen Modells stellen die Änderung der abhängigen Variable dar, in diesem Fall die Druckfestigkeit von Beton. Diese Änderung erfolgt für eine Einheitsänderung der unabhängigen Variablen, unter der Annahme, dass alle anderen Variablen konstant bleiben.

Positive Koeffizienten deuten auf eine Erhöhung der Beton-Druckfestigkeit bei Zunahme der Variablen hin, während negative Koeffizienten eine Verringerung anzeigen. Der Intercept-Wert repräsentiert die geschätzte Beton-Druckfestigkeit, wenn alle unabhängigen Variablen den Wert null annehmen.

Daraus ergibt sich zusammen mit Gleichung 5.1 die Regressionsgerade für das Modell.

Tabelle 9.: Koeffizienten des linearen Regressionsmodells.

Merkmal (β_j)	Koeffizient
Intercept (β_0)	4.36596
cement	0.75091
blast	0.06610
ash	0.02683
water	-0.92315
superplasticizer	0.06410
coarse	0.08554
fine	-0.31901
age	0.29090

Quelle: Eigene Darstellung, A.3.

Die Modellmetriken, dargestellt in Tabelle 10, geben Auskunft über die Vorhersagegenauigkeit und die Anpassungsgüte des Modells.

Der mittlere absolute Fehler (MAE) von 0.19 zeigt an, dass die Vorhersagen des Modells im Durchschnitt um 0.19 Einheiten vom tatsächlichen Wert abweichen. In Relation zu dem betrachteten Wertebereich (vgl. Abbildung 6) der Zielvariablen stellt dies einen kleinen Prozentsatz der möglichen maximalen Abweichung dar, was auf eine zufriedenstellende Vorhersagegenauigkeit des Modells hindeutet.

Der mittlere quadratische Fehler (MSE) von 0.06 misst die durchschnittliche quadratische Abweichung der Vorhersagen vom tatsächlichen Wert. Ein niedriger MSE-Wert wie 0.06 deutet auf eine geringe Fehlervarianz und somit auf eine hohe Konsistenz der Modellvorhersagen hin.

Die Wurzel des mittleren quadratischen Fehlers (engl. *Root Mean Squared Error*, RMSE) von 0.24, bietet eine noch präzisere Darstellung der durchschnittlichen Fehlergröße. Da größere Fehler stärker gewichte werden, deutet ein Wert von 0.24 darauf hin, dass das Modell tendenziell genaue Vorhersagen liefert, wobei größere Abweichungen seltener auftreten.

Die R^2-Werte für Training und Test von 0.7852 bzw. 0.8155 sind ein Maß für die Güte der Modellanpassung. Ein R^2-Wert nahe 1 deutet auf eine hohe Erklärungskraft des Modells hin. In diesem Fall erklären die R^2-Werte von 0.7852 und 0.8155, dass das Modell einen signifikanten Anteil der Varianz in den Daten erfasst, was auf eine effektive Modellierung der Zusammenhänge in den Daten hindeutet.

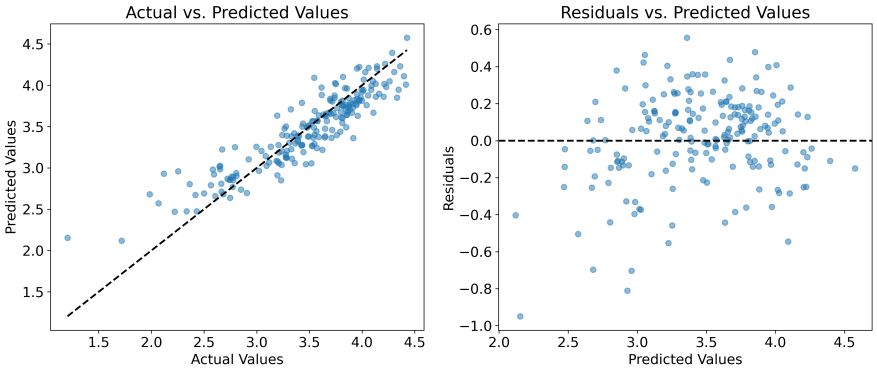
Tabelle 10.: Modellmetriken des linearen Regressionsmodells.

Metrik	Wert
Mean Absolute Error (MAE)	0.19
Mean Squared Error (MSE)	0.06
Root Mean Squared Error (RMSE)	0.24
Training Score (R^2)	0.7852
Test Score (R^2)	0.8155

Quelle: Eigene Darstellung, A.3.

Darüber hinaus wurden die tatsächlichen gegen die vorhergesagten Werte in Abbildung 6 visualisiert, welche eine allgemeine Einschätzung der Modellgenauigkeit ermöglichen. Ein weiterer Aspekt sind die Residuen des Modells. Die Residuen, also die Differenzen zwischen den tatsächlichen und vorhergesagten Werten, sollten idealerweise zufällig um null verteilt sein und keine Muster aufweisen, die auf eine Verletzung der Modellannahmen hindeuten könnten. Die Residuen aus Abbildung 6 erfüllen diese wünschenswerten Eigenschaften.

Abbildung 6.: Residuenanalyse: Beziehung zwischen Vorhersagen und Abweichungen.



Quelle: Eigene Darstellung, A.3.

6.2. Interpretation der Koeffizienten & Permutation der Merkmalrelevanz

Die Koeffizienten beschreiben eine bedingte Assoziation. Das bedeutet, sie quantifizieren die Variation der Druckfestigkeit, wenn eine bestimmte unabhängige Variable verändert wird, während alle anderen unabhängigen Variablen konstant gehalten werden.

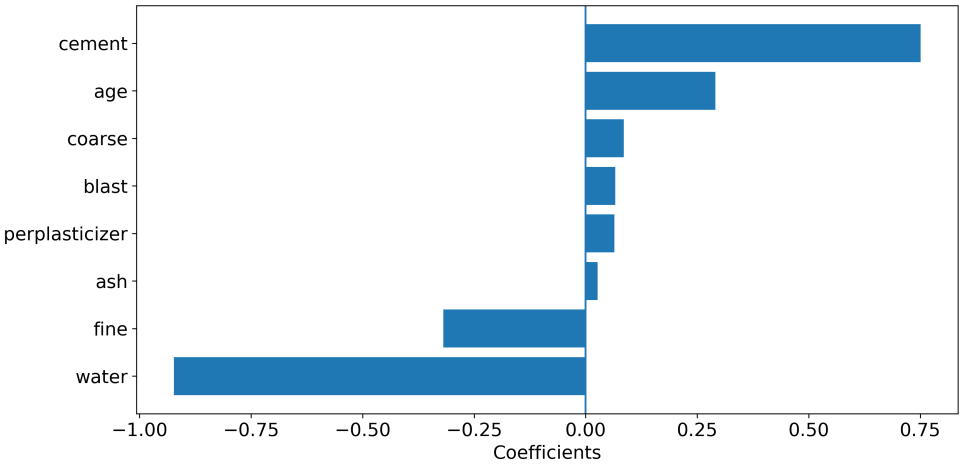
Die Koeffizienten sollten somit nicht als marginale Beiträge betrachtet werden. Das bedeutet, sie beschreiben nicht die Beziehung zwischen den Variablen unabhängig von anderen Einflussfaktoren. Stattdessen zeigen sie, wie sich die Druckfestigkeit ändert, wenn eine bestimmte unabhängige Variable variiert wird, während alle anderen konstant gehalten werden.

Abbildung 7 zeigt die Koeffizienten des Regressionsmodells. Die Stärke des Einflusses einer unabhängigen Variable auf die abhängige Variable hängt von der Größe der Merkmalsausprägung ab. Ob ein bestimmtes Merkmal einen großen oder kleinen Einfluss auf die abhängige Variable hat, hängt von den spezifischen Werten der Merkmale und der Streuung der Merkmalsausprägungen ab.

Die logarthimische Transformation der Merkmale hat dieses Problem bereits teilweise gelöst, indem sie die Skalierung der Daten angepasst hat und alle Merkmale auf eine einheitliche Skala transformiert hat. Daraus lassen sich erste Schlüsse der Merkmalsrelevanz ableiten. Im Vergleich zu allen anderen Koeffizienten tragen die Merkmale Flugasche, Superplastifikator, Hochofenschlacke und grober Zuschlag nur wenig zur abhängigen Variable bei. Eine Erhöhung von Zement führt zu einer Steigerung der Druckfestigkeit. Im Gegensatz dazu führt eine Erhöhung des Wasseranteils zu einer Verringerung der Druckfestigkeit, immer unter der Vorraussetzung, das alle anderen Merkmale konstant bleiben.

Abbildung 8 zeigt einen Effektplot für die erste Beobachtung im Testdatensatz. Die-

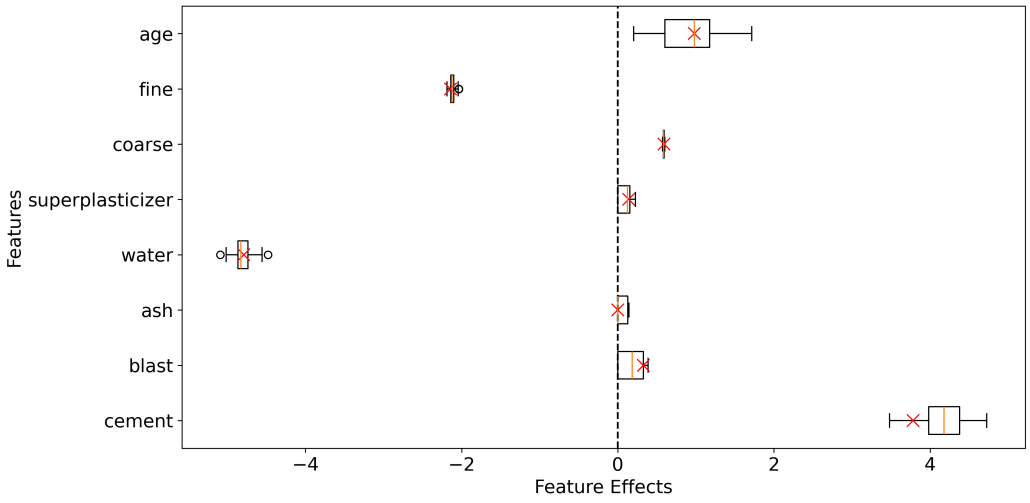
Abbildung 7.: Koeffizienten des linearen Regressionsmodells.



Quelle: Eigene Darstellung, A.3.

ser ermöglicht es, den individuellen Effekt eines Merkmals auf die Vorhersage in Relation zur gesamten Verteilung aller Beobachtungen zu veranschaulichen. Die graphische Darstellung zeigt die Auswirkungen der Merkmale auf die Vorhersage für diese spezielle Beobachtung. Jedes Merkmal wird durch ein rotes Kreuz markiert. Dieses Kreuz stellt den Effekt des Merkmals auf die Vorhersage dar, welcher sich aus dem Produkt des Koeffizienten und der Merkmalsausprägung ergibt. Die horizontale Position des Kreuzes zeigt den Wert des Effekts, während die vertikale Position das entsprechende Merkmal repräsentiert.

Abbildung 8.: Effektplot für Beoabchtung $x^{(0)}$ der Testmenge.



Quelle: Eigene Darstellung, A.3.

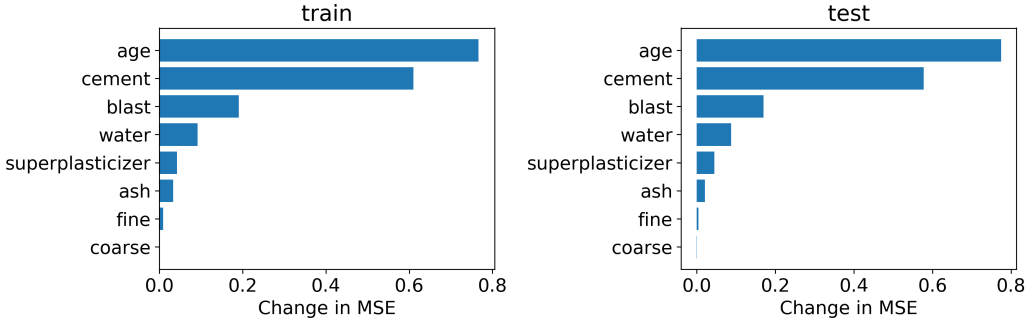
Der Effektplot ist ein nützliches Werkzeug, um zu verstehen, wie sich Änderungen in den Merkmalswerten auf die Vorhersage auswirken und wie dieser individuelle Effekt im Vergleich zur gesamten Datenverteilung steht. Für ein vollständiges Verständnis

der Bedeutung jedes Merkmals im gesamten Datensatz, insbesondere im Kontext der Vorhersage der Druckfestigkeit von Beton, erweist sich jedoch eine globale Betrachtung des Einflusses der Merkmale als unerlässlich.

Ein bewährtes Verfahren zur Bestimmung der Merkmalsrelevanz ist die sogenannte Permutation der Merkmalrelevanz (engl. *Permutation Feature Importance*, PFI). Diese Methode misst die Veränderung im Vorhersagefehler des Modells nach der zufälligen Vertauschung der Werte eines bestimmten Merkmals. Ein Merkmal gilt als wichtig, wenn die Vertauschung seiner Werte den Modellfehler erhöht, da das Modell stark auf dieses Merkmal für seine Vorhersagen angewiesen ist. Umgekehrt wird ein Merkmal als unwichtig betrachtet, wenn die Vertauschung seiner Werte den Modellfehler unverändert lässt, da das Modell das Merkmal für seine Vorhersagen ignoriert [Mol22, S. 157].

Abbildung 9 zeigt die Merkmalsrelevanz auf der Test- und Trainingsmenge anhand der Veränderung des mittleren quadratischen Fehlers des Modells.

Abbildung 9.: Permutation der Merkmalrelevanz.



Quelle: Eigene Darstellung, A.3.

Es ist festzustellen, dass die Merkmale grober Zuschlag, feiner Zuschlag und Flugasche keine erhebliche Veränderung im mittleren quadratischen Fehler des Modells hervorrufen und diese somit im Vergleich zu den anderen Merkmalen als eher irrelevant angesehen werden können.

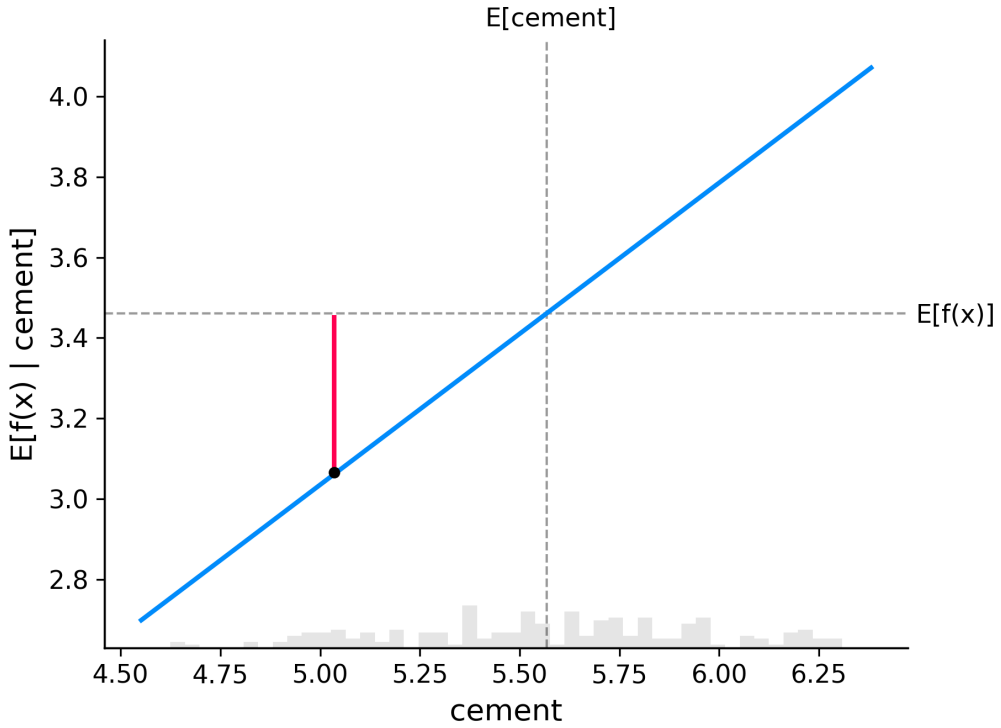
6.3. Interpretation mit SHAP

SHAP-Werte berücksichtigen die Wechselwirkungen zwischen den Merkmalen und quantifizieren den Beitrag jedes Merkmals zur Abweichung der Prognose im Mittel. Dies ermöglicht eine präzisere Interpretation, da die SHAP-Werte den Einfluss eines Merkmals auf die Vorhersage unter Berücksichtigung aller anderen Merkmale anzeigen. Durch die Verwendung von SHAP-Werten wird es möglich, die Beiträge der einzelnen Merkmale zur Vorhersageleistung des Modells nicht nur auf globaler Ebene, sondern auch auf lokaler, individueller Ebene zu verstehen.

6.3.1. Lokale Interpretation

Die lokale Interpretation konzentriert sich auf das Verständnis der Vorhersagen für eine einzelne Beobachtung aus dem Datensatz.

Abbildung 10.: SHAP Partial Dependence Plot für Beobachtung $x^{(0)}$ der Testmenge.



Quelle: Eigene Darstellung, A.3.

Im Partial Dependence Plot der Abbildung 10 wird die Beziehung zwischen dem Merkmal Zement und der Zielvariable für die spezifische Beobachtung $x^{(0)}$ aus der Testmenge visualisiert. Das Histogramm zeigt die Verteilung der konkreten Merkmalsausprägungen über die Gesamtheit der Daten. Die blaue Linie im Diagramm repräsentiert die modellierte Abhängigkeit der Vorhersage vom Merkmal Zement, unter Konstanthaltung aller anderen Merkmale. Der schwarze Punkt markiert die tatsächliche Ausprägung des Merkmals Zement (5.035) für die betrachtete Beobachtung und die korrespondierende Vorhersage des linearen Regressionsmodells. Die rote Linie illustriert die marginale Abweichung der Vorhersage von der durchschnittlichen Modellprognose $\mathbb{E}[f(X)] = 3.457$, ausgehend vom beobachteten Wert von Zement nach Gleichung 4.1:

$$\begin{aligned} \varphi^{(0)}_{\text{cement}}(f, x) &= \beta_{\text{cement}}(x^{(0)}_{\text{cement}} - \mathbb{E}[X_{\text{cement}}]) \\ &= 0.75091 \cdot (5.035 - 5.568) \\ &\approx -0.40 \end{aligned} \tag{6.1}$$

Codeausschnitt 6.1 zeigt die manuelle Berechnung der für $\varphi^{(0)}_{\text{cement}}(f, x)$, basierend auf den zur Verfügung stehenden Daten für das Merkmal Zement[10] .

Codeauschnitt 6.1: Berechnen der SHAP-Werte für ein Merkmal und eine Beobachtung, A.3.

```python
def calculate_shap_contribution(coef: pd.DataFrame, shap_values: shap.Explanation, feature: str, idx: int):
    """
    Calculate the average contribution of the feature to the model prediction.

    Args:
        coef: A pandas Dataframe containing the coefficients of the linear regression model.
        shap_values: A SHAP Explanation object containing the SHAP values for the model features.
        feature: The name of the feature.
        idx: The index of obersavtion to calculate the value for.

    Returns:
    - The average SHAP contribution for the feature.
    """
    coef = coef.loc[feature].values[0]
    feature_values = shap_values[:, feature].data
    mean = np.mean(feature_values)
    shap_contribution = coef * (feature_values[idx] - mean)
    return shap_contribution

print(calculate_shap_contribution(coef=coef, shap_values=shap_values, feature="cement", idx=0))
```

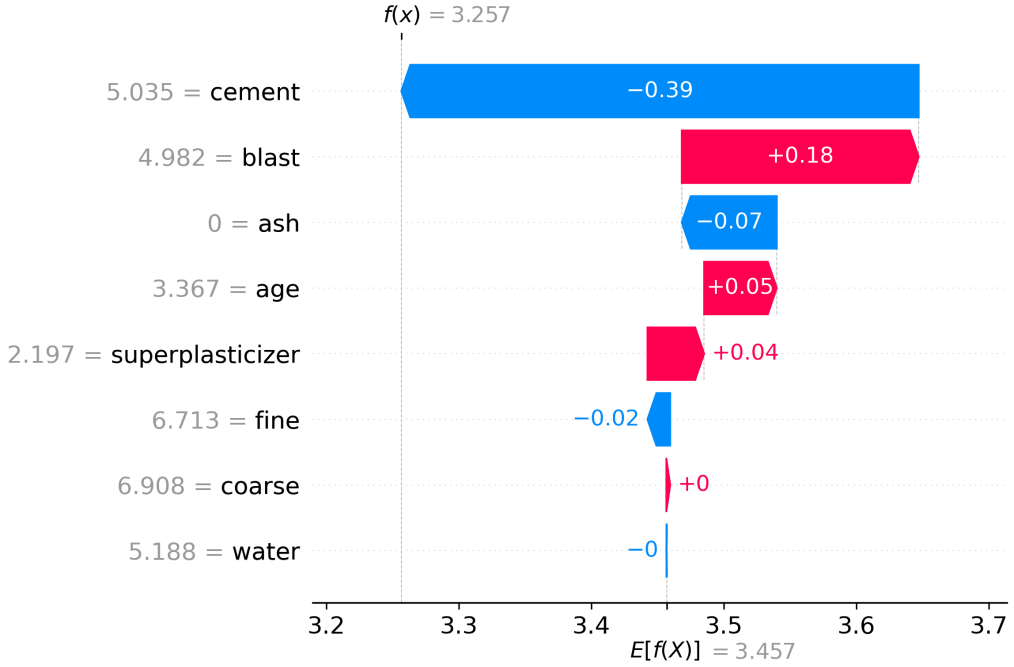
Die Anordnung des schwarzen Punktes entlang der funktionalen Beziehung gibt den spezifischen Wert von Zement an und reflektiert, wie dieser Wert in den Kontext des gesamten Wertebereichs dieses Merkmals eingeordnet wird. Die vertikale Distanz zwischen der durchschnittlichen Vorhersage (dargestellt durch die horizontale gestrichelte Linie) und dem Punkt auf der funktionalen Abhängigkeit (blaue Linie) zeigt den Einfluss des Merkmals Zement auf die individuelle Vorhersage im Vergleich zum Modellmittelwert. Dieser Magnitude der roten Linie verkörpert den SHAP-Wert, den marginalen Beitrag des Merkmals Zement zur Prognoseabweichung für die ausgewählte Beobachtung.

Während der Partial Dependence Plot einen wertvollen Einblick in die Modellabhängigkeit von einzelnen Merkmalen bietet, ist für ein umfassendes Verständnis der Modellvorhersage eine ganzheitliche Betrachtung aller Merkmale notwendig. Der SHAP Waterfall Plot adressiert diese Notwendigkeit, indem er eine kumulative Darstellung

[10] Die Anwendung des SHAP-Explainers, insbesondere im Kontext eines Train-Test-Splits, stößt aufgrund der aktuellen Implementierung im SHAP-Python-Paket auf Herausforderungen. Die SHAP-Werte werden auf Basis des Datensatzes X berechnet, was bei der Verwendung von Teilmengen wie X_{test} zu geringfügigen Abweichungen führen kann, wie Berechnung 6.1 zeigt. Dieses Verhalten ist insbesondere in Diskussionen auf GitHub zu erkennen, wie beispielsweise in dem Issue https://github.com/shap/shap/issues/3456 diskutiert. Eine Erörterung dieses Problems in Bezug auf die interne Logik und die Verwendung des Pakets findet sich in einem von mir verfassten Stack Overflow Beitrag, verfügbar unter https://stackoverflow.com/questions/77820555/shap-partial-dependence-plot-misalignment-with-train-test-split-in-linear-regres.

aller marginalen Beiträge liefert. Jedes Merkmal wird in Form einer Sequenz von Beiträgen visualisiert, beginnend mit dem Basiswert der Vorhersage, welcher durch die Addition oder Subtraktion der individuellen Merkmalsbeiträge schrittweise zur finalen Vorhersage modifiziert wird.

Abbildung 11.: SHAP Waterfall Plot für Beobachtung $x^{(0)}$ der Testmenge.



Quelle: Eigene Darstellung, A.3.

In Abbildung 11 ist ein Waterfall Plot der ersten Beobachtung $x^{(0)}$ dargestellt, der die Zerlegung einer einzelnen Modellvorhersage zeigt. Der Plot beginnt mit dem Basiswert $\mathbb{E}[f(X)] = 3.457$, der durchschnittlichen Vorhersage des Modells.

Von diesem Wert ausgehend, illustrieren die Balken, wie jede Merkmalausprägung – angezeigt durch die grauen Zahlen entlang der y-Achse – die Vorhersage $f(x_j^{(0)})$ beeinflusst. So steigert beispielsweise die Hochofenschlacke mit einem Wert von 4.982 die Vorhersage um +0.18, wohingegen Zement mit einem Wert von 5.035 die Vorhersage um −0.39 verringert.

Rote Balken repräsentieren Merkmale, die die Vorhersage erhöhen, während blaue Balken solche darstellen, die sie senken. Die Größe jedes Balkens zeigt das Ausmaß des jeweiligen Beitrags, und die abschließende Vorhersage $f(x) = 3.257$ wird am Ende der Kette dieser Effekte erreicht.

Kleine positive und negative Beiträge der Merkmale wie feiner Zuschlag (6.713), grober Zuschlag (6.908) und dem Anteil von Wasser (5.188) in der Zusammensetzung des Betons zeigen, wie feingranulare Anpassungen der Merkmalsausprägungen die Vorhersage nicht verändern, leicht erhöhen oder senken können.

Für die Beobachtung $x^{(0)}$ führt die kumulative Abweichung der Merkmal-Effekte vom Basiswert $\mathbb{E}[f(X)] = 3.457$ zu einem tatsächlichen Modelloutput von $f(x) = 3.257$, was eine Differenz von -0.2 zwischen der durchschnittlichen Vorhersage und der spezifischen Vorhersage für diese Beobachtung offenlegt. Diese Differenz entspricht der Summe aller SHAP-Werte für diese konkrete Beobachtung [Mol23, S. 52f].

Da die Zielgröße einer logarithmischen Transformation unterzogen wurde, muss diese für die Interpreation wieder rückgängig gemacht werden. Dies bedeutet, dass der tatsächliche erwartete Wert der Druckfestigkeit der Exponentialfunktion des prognostizierten Wertes entspricht, also $e^{3.457} \approx 34.71$ MPa. Dieser Rücktransformationsprozess ist notwendig, um die Modellprognosen in der ursprünglichen Skala der Zielvariablen zu interpretieren. Dies gilt darüberhinaus sowohl für die einzelnen SHAP-Werte, als auch für die konkrete Vorhersage $f(x) = 3.257$. Die prognostizierte Durckfestigkeit für die Beobachtung $x^{(0)}$ beträgt folglich $e^{3.257} \approx 25.97$ MPa.

Dies ermöglicht eine detaillierte Analyse, wie das Modell zu einer bestimmten Vorhersage kommt, und hilft dabei, die Beiträge und Interaktionen zwischen verschiedenen Merkmalen zu verstehen.

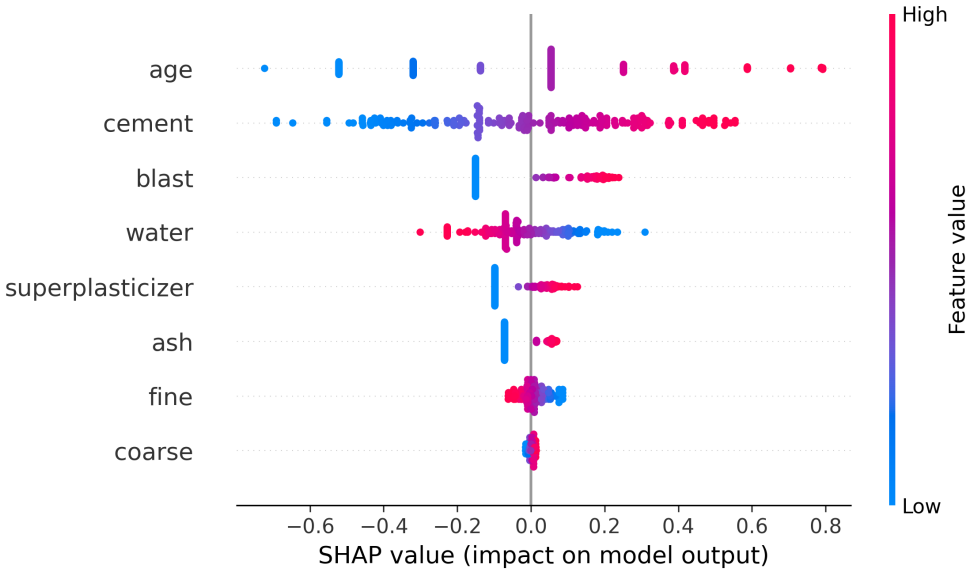
Die lokale Interpretation mittels SHAP-Werten ermöglicht zwar eine präzise Erklärung der Modellvorhersagen für individuelle Beobachtungen, jedoch stellt sich bei einer solchen Betrachtung das Problem der fehlenden Generalisierbarkeit. Lokale Analysen können dazu führen, dass spezifische Merkmal-Kontributionen überinterpretiert werden, ohne die übergeordneten Muster und Einflüsse zu berücksichtigen, die das Modellverhalten im gesamten Datensatz charakterisieren. Eine globale Interpretation ist daher erforderlich, um die Konsistenz und Zuverlässigkeit des Modells über verschiedene Beobachtungen hinweg zu erfassen.

6.3.2. Globale Interpretation

Der SHAP Beeswarm Plot in Abbildung 12 bietet eine globale Sicht auf die Modellvorhersagen, indem er die Verteilung der SHAP-Werte für jedes Merkmals über alle Beobachtungen hinweg darstellt. Jeder Punkt repräsentiert eine Beobachtung aus dem Datensatz. Die Farbe der Punkte zeigt die Merkmalsausprägungen an: hohe Werte in Rot und niedrige Werte in Blau. Die Position auf der x-Achse gibt den Einfluss des Merkmals auf die Modellvorhersage an. Positive SHAP-Werte (rechts von der Nulllinie) zeigen eine Erhöhung der Vorhersage an, während negative Werte (links von der Nulllinie) eine Verringerung bedeuten.

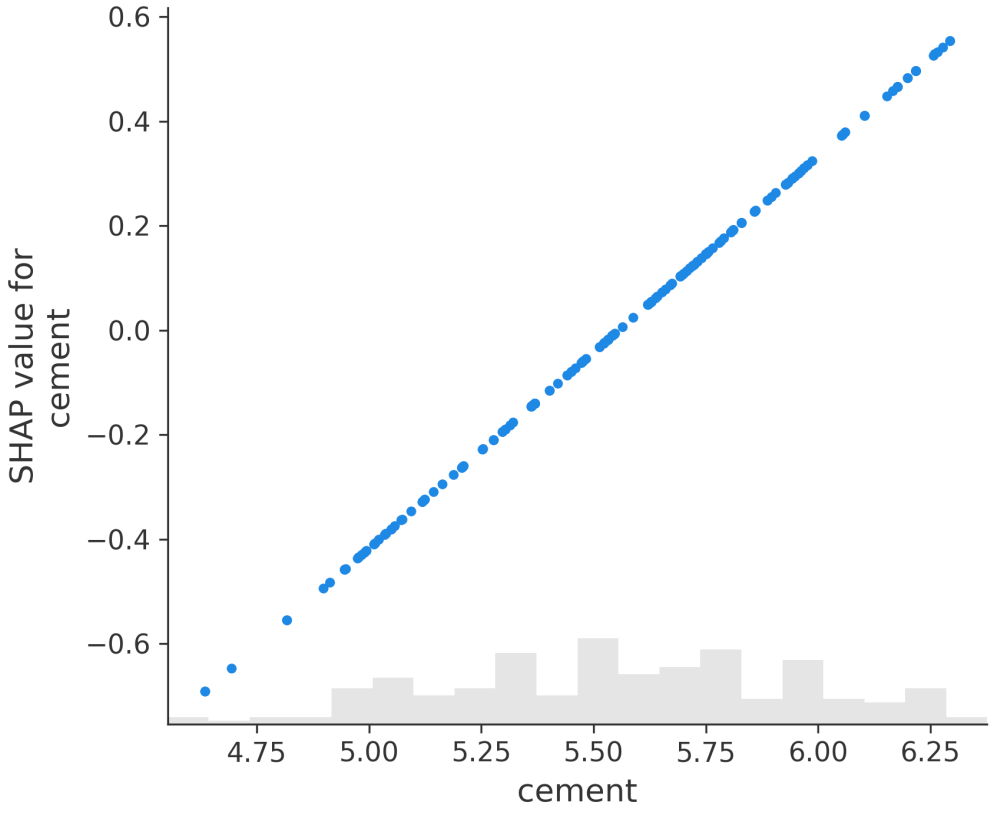
Das Merkmal Alter zeigt eine hohe Variabilität in seinem Einfluss auf die Modellvorhersage. Höhere Werte von Zement sind mit einer Zunahme der Vorhersage (positive SHAP-Werte) assoziiert, was durch die rechtsseitigen Punkte in der Grafik dargestellt wird. Niedrigere Werte führen hingegen zu einer geringeren Vorhersage. Diese Streuung der Punkte zeigt, dass die Auswirkung von Zement auf die Vorhersage stark von seiner quantitativen Ausprägung abhängt, wie Abbildung 13 verdeutlicht.

Abbildung 12.: SHAP Beeswarm Plot.



Quelle: Eigene Darstellung, A.3.

Abbildung 13.: SHAP Scatter Plot.

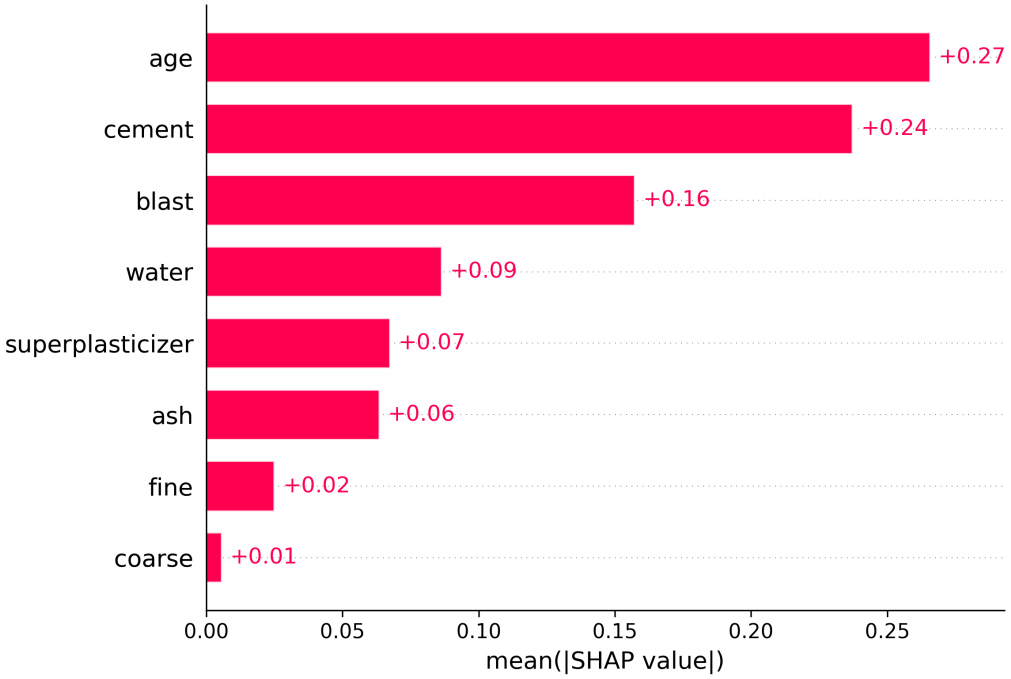


Quelle: Eigene Darstellung, A.3.

Der Scatter Plot zeigt die SHAP-Werte für Zement bezüglich ihrer quantitivaten Ausprägung über alle Daten. Steigende Merkmalsausprägungen von Zement führen zu größeren SHAP-Werten und somit zu einer Erhöhung der Modellprognose, was die ersten Analysen der Korrelationsmatrix aus Abbildung 5 bestätigen. Der Einfluss des Wasseranteils ist invers. Eine Erhöhung des Wasseranteils resultiert in geringeren SHAP-Werten und reduziert somit die Druckfestigkeit.

Diese Darstellungen ermöglichen es, die Merkmale zu identifizieren, die den größten Einfluss auf das Modell haben und wie dieser Einfluss über unterschiedliche Beobachtungen variiert.

Abbildung 14.: SHAP Bar Plot.



Quelle: Eigene Darstellung, A.3.

Der SHAP Bar Plot in Abbildung 14 illustriert die durchschnittliche Auswirkung jedes Merkmals auf das Modell, gemessen an der absoluten Größe der SHAP-Werte über alle Beobachtungen hinweg. Die Balken zeigen die durchschnittlichen Beiträge der Merkmale zur Vorhersage: Je länger der Balken, desto größer ist der Einfluss des jeweiligen Merkmals. Hier ist das Merkmal Alter mit dem höchsten durchschnittlichen SHAP-Wert (+0.27) das einflussreichste Merkmal, was auf eine starke positive Beziehung zur Zielvariablen hinweist. Die weiteren Merkmale folgen in absteigender Reihenfolge ihrer Bedeutung.

Es fällt auf, dass eine Übereinstimmung in der Reihenfolge der Merkmalsrelevanz zwischen der Permutation der Merkmalrelevanz aus Abbildung 9 und den SHAP-Werten aus Abbildung 14 besteht, obwohl die zugrundeliegenden Interpretationen dieser beiden Methoden deutlich verschieden sind.

Die Permutation der Merkmalrelevanz konzentriert sich darauf, die Veränderungen im Vorhersagefehler des Modells, speziell den mittleren quadratischen Fehler, zu messen, die eintreten, wenn die Werte eines Merkmals zufällig vertauscht werden [Mol22, S. 157]. Diese Methode gibt Aufschluss darüber, wie stark das Modell auf das jeweilige Merkmal für seine Vorhersagegenauigkeit angewiesen ist. Ein wesentlicher Aspekt dieser Methode ist, dass sie nicht direkt die Wechselwirkungen zwischen den Merkmalen berücksichtigt. Sie zeigt vielmehr, wie wichtig ein Merkmal isoliert für die Gesamtleistung des Modells ist.

Im Gegensatz dazu bieten die SHAP-Werte einen tieferen Einblick in die Beiträge jedes Merkmals zur Vorhersageleistung des Modells. Sie quantifizieren den Einfluss eines Merkmals auf die Abweichung der Prognose vom Basiswert, also der durchschnittlichen Vorhersage des Modells. Hierbei wird nicht nur die individuelle Wichtigkeit jedes Merkmals hervorgehoben, sondern auch deren Wechselwirkungen mit anderen Merkmalen berücksichtigt. Diese Methodik ermöglicht es, sowohl eine globale als auch eine lokale Perspektive auf die Modellvorhersagen zu werfen. Während die globale Interpretation durchschnittliche Auswirkungen aller Merkmale aufzeigt, erlaubt die lokale Sichtweise, die Vorhersagen für einzelne Beobachtungen präzise zu erklären.

Diese unterschiedlichen Herangehensweisen und Interpretationen der Merkmalsrelevanz, einerseits durch die Permutation der Merkmalrelevanz und andererseits durch die SHAP-Werte, verdeutlichen die Komplexität und die Tiefe der Analyse, die für ein umfassendes Verständnis von Vorhersagemodellen erforderlich ist. Beide Methoden ergänzen sich gegenseitig und tragen dazu bei, ein vollständigeres Bild der Dynamiken innerhalb des Modells zu zeichnen.

SHAP-Werte bieten für die Prognose der Druckfestigkeit von Beton einen signifikanten Vorteil, da sie eine gerechte Verteilung des Vorhersagebeitrags über alle Merkmale gewährleisten. Diese gerechte Verteilung ist eine der Kernstärken der SHAP-Werte, die sie von anderen Methoden abhebt.

Die Druckfestigkeit von Beton ist das Ergebnis einer komplexen Interaktion verschiedener Materialkomponenten und Eigenschaften wie Zementgehalt, Wasser-Zement-Verhältnis, Zuschlagstoffe und Alter des Betons. Jede dieser Komponenten trägt unterschiedlich zur Endfestigkeit bei. Ihre Effekte können nicht isoliert betrachtet werden, da sie aufeinander abgestimmt werden müssen [NK21, S. 2].

Andere Methoden, wie beispielsweise die Betrachtung von Regressionskoeffizienten, geben zwar Aufschluss über die Richtung und Stärke des Zusammenhangs zwischen Merkmalen und der Zielvariable, vernachlässigen jedoch die Interaktionseffekte zwischen den Merkmalen.

In der Praxis bedeutet dies für die Prognose der Druckfestigkeit von Beton, dass SHAP-Werte eine präzise Zuordnung der Einflussstärke jedes Bestandteils und Ver-

arbeitungsmerkmals erlauben. Da die Festigkeit von Beton von einer Vielzahl von Faktoren abhängt, ermöglicht die granulare Aufschlüsselung der SHAP-Werte eine detailreiche Einsicht, welche Komponenten optimiert werden sollten, um die gewünschten Eigenschaften des Betons zu erreichen.

Im Kontext von industriellen Anwendungen und Forschung, wo Entscheidungen auf Grundlage der Modellvorhersagen getroffen werden, gewährleisten SHAP-Werte somit eine transparente und objektive Grundlage. Dies ist nicht nur für die Entwicklung von Betonmischungen von Bedeutung, sondern auch für die Einhaltung von Bauvorschriften und die Gewährleistung der Sicherheit. Durch die Verwendung von SHAP-Werten kann die Forschung im Bereich der Materialwissenschaften fundierter und zielgerichteter gestaltet werden, was zu einer effizienteren und effektiveren Materialentwicklung führt.

7. Fazit

Diese Arbeit hat die Anwendung und Interpretation von SHAP-Werten in der linearen Regressionsanalyse zur Vorhersage der Druckfestigkeit von Beton umfassend untersucht. SHAP-Werte, als ein fortschrittlicher Ansatz zur Modellinterpretation, haben sich als besonders wertvoll erwiesen, indem sie eine tiefere und nuanciertere Einsicht in die Beiträge einzelner Merkmale zur Modellvorhersage bieten. Im Vergleich zu herkömmlichen Interpretationsmethoden, wie der Analyse der Regressionskoeffizienten und der Permutation der Merkmalrelevanz, bieten SHAP-Werte mehrere Vorteile.

Einer der Hauptvorteile von SHAP-Werten liegt in ihrer Fähigkeit, sowohl globale als auch lokale Interpretationen zu ermöglichen. Sie erlauben es, den Einfluss einzelner Merkmale auf spezifische Vorhersagen zu verstehen, während gleichzeitig ein Überblick über deren Bedeutung im gesamten Modell gegeben wird. Darüber hinaus berücksichtigen SHAP-Werte die Interaktionen zwischen den Merkmalen und bieten somit eine ganzheitliche Sicht auf die Modellvorhersagen. Dies ist besonders relevant in komplexen Datenstrukturen, wo Merkmalsinteraktionen eine signifikante Rolle spielen.

Ein weiterer zentraler Vorteil der SHAP-Werte liegt in ihrer umfassenden Erklärungskraft. Im Gegensatz zur Permutation der Merkmalrelevanz, die sich darauf konzentriert, wie sich der Vorhersagefehler des Modells verändert, wenn die Werte eines Merkmals zufällig vertauscht werden, bieten SHAP-Werte eine detaillierte Aufschlüsselung des Beitrags jedes einzelnen Merkmals zur Vorhersage. Diese Eigenschaft der SHAP-Werte ist von Vorteil, da sie eine faire Aufteilung der Vorhersageabweichung unter allen Merkmalsausprägungen gewährleistet – bekannt als die Effizienzeigenschaft der Shapley-Werte. Diese faire Aufteilung ist dann wichtig, wenn eine vollständige und transparente Erklärbarkeit erforderlich ist.

Trotz dieser Stärken ist es wichtig, auch die Grenzen der SHAP-Methode zu erkennen. Eine der Herausforderungen bei der Anwendung von SHAP-Werten ist die Interpretation der Ergebnisse, insbesondere wenn es um komplexe Modelle mit einer großen Anzahl von Merkmalen und komplizierten Interaktionen geht. Die Interpretation kann schnell überwältigend werden und erfordert ein tiefes Verständnis der zugrundeliegenden Daten und des Modells. Darüber hinaus kann die Berechnung von SHAP-Werten insbesondere bei großen Datensätzen rechenintensiv sein, was praktische Einschränkungen mit sich bringt.

Ein Nachteil ist die potenzielle Verzerrung, die durch die Stichprobenabhängigkeit

der SHAP-Werte entstehen kann. SHAP-Werte basieren auf der Annahme, dass die Daten, auf denen das Modell trainiert wurde, repräsentativ für die zugrunde liegende Population sind. Wenn diese Annahme nicht erfüllt ist, können die SHAP-Werte irreführende Interpretationen liefern.

Zusammenfassend lässt sich sagen, dass SHAP-Werte eine bedeutende Erweiterung der Möglichkeiten zur Interpretation von linearen Modellen darstellen. Sie bieten tiefergehende Einblicke in die Funktionsweise von linearen Modellen und ermöglichen eine präzisere und umfassendere Analyse der Merkmalsrelevanz und -interaktionen. Dennoch ist es entscheidend, ihre Limitationen zu verstehen und sie als einen Teil eines umfassenden Prozesses der Modellinterpretation und -validierung zu betrachten.

Die Anwendung von SHAP-Werten ist keineswegs auf lineare Modelle beschränkt. Vielmehr ermöglicht das shap-Paket mit verschiedensten Estimatoren – darunter TreeSHAP für baumbasierte Modelle und DeepSHAP für neuronale Netzwerke – den Einsatz dieser interpretativen Methode über ein breites Spektrum nichtlinearer Modelle hinweg. Diese Berechnungsmethoden sind entweder modellspezifisch oder modellagnostisch konzipiert, um den unterschiedlichen Anforderungen der SHAP-Wertberechnung in komplexeren Modellstrukturen gerecht zu werden und bieten somit eine flexible Grundlage für tiefere Einblicke in die Funktionsweise verschiedenster Algorithmen [Mol23, S. 40f].

Literaturverzeichnis

[AFSS19] Encarnación Algaba, Vito Fragnelli, and Joaquín Sánchez-Soriano. Handbook of the shapley value. 2019. URL: https://api.semanticscholar.org/CorpusID:213768429.

[FWL+14] Changyong Feng, Hongyue Wang, Naiji Lu, Tian Chen, Hua He, Ying Lu, and Xin M Tu. Log-transformation and its implications for data analysis. *Shanghai Archives of Psychiatry*, 26(2):105–109, 2014. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4120293/pdf/sap-26-02-105.pdf.

[LL17] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

[Mol22] Christoph Molnar. *Interpretable machine learning: A guide for making Black Box models explainable*. Chistoph Molnar c/o Mucbook Clubhouse, Heidi Seibold, 2 edition, 2022.

[Mol23] Christoph Molnar. *Interpreting machine learning models with SAP A guide with python examples and theory on Shapley Values*. Chistoph Molnar c/o MUCBOOK, 1 edition, 2023.

[NK21] K. Nandhini and J. Karthikeyan. The early-age prediction of concrete strength using maturity models: a review. *Journal of Building Pathology and Rehabilitation*, 6(1):7, 2021. URL: https://www.researchgate.net/profile/Nandhini-Kumar/publication/348238311_The_early-age_prediction_of_concrete_strength_using_maturity_models_a_review/links/603c75b84585158939d98b5f/The-early-age-prediction-of-concrete-strength-using-maturity-models-a-review.pdf.

[RWB+22] Benedek Rozemberczki, Lauren Watson, Péter Bayer, Hao-Tsung Yang, Olivér Kiss, Sebastian Nilsson, and Rik Sarkar. The shapley value in machine learning. In Lud De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5572–5579. International Joint Conferences on Artificial Intelligence

Organization, 7 2022. Survey Track. URL: https://doi.org/10.24963/ijcai.2022/778.

[Sha53] L. S. Shapley. *17. A Value for n-Person Games*, pages 307–318. Princeton University Press, Princeton, 1953. URL: https://doi.org/10.1515/9781400881970-018.

[Yeh07] I-Cheng Yeh. Concrete Compressive Strength. UCI Machine Learning Repository, 2007. URL: https://doi.org/10.24432/C5PK67.

Eidesstattliche Erklärung

Ich versichere an Eides statt, dass ich die vorstehende Arbeit selbständig und ohne fremde Hilfe angefertigt und mich anderer als der im beigefügten Verzeichnis angegebenen Hilfsmittel nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen übernommen wurden, sind als solche kenntlich gemacht. Alle Internetquellen sind der Arbeit beigefügt.

Des Weiteren versichere ich, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und dass die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

München, 16. März 2024

Simon Symhoven

A. Quellcode

A.1. requirements.txt

```
pandas
matplotlib
scikit-learn
numpy
shap
seaborn
statsmodels
mlxtend
requests
xlrd
```

A.2. charts.py

```python
import matplotlib.pyplot as plt
from typing import List

plt.rcParams.update({
    'text.usetex': True,
    'text.latex.preamble': r'\usepackage{amsfonts}',
    'font.size': 15
})

def plot_waterfall(expected_value_model: float, observation_values: List[float],
                   observation_labels: List[str], contributions: List[float],
                   name: str, index: int):
    """
    Creates a waterfall chart plot that visualizes the expected model value,
    observation values, their contributions, and labels.

    Parameters:
        - expected_value_model (float): The expected model value.
        - observation_values (List[float]): A list of observation values.
        - observation_labels (List[str]): A list of labels associated with
                                          the observation values.
        - contributions (List[float]): A list of contributions representing
                                       changes from observation values to the
                                       model value.
        - name (str): The name of the plot and the file for saving.
        - index (int): The index for the observation.
    """

    labels = [f'{value} = {label}' for (value, label)
              in zip(observation_values, observation_labels)]
    waterfall = [expected_value_model]
    for contribution in contributions:
        waterfall.append(waterfall[-1] + contribution)

    _, ax = plt.subplots(figsize=(10, 3))

```

```
    for idx, (contribution, start) in enumerate(
        zip(contributions, waterfall[:-1])
    ):
        ax.barh(idx, contribution, left=start, color='blue'
                if contribution >= 0 else 'red', height=0.5, alpha=0.8)
        text_x = start + contribution / 2
        ax.text(text_x, idx, contribution, va='center', ha='center', fontsize=12,
    color='white')

    l1 = r'$\mathbb{E}(f(X)) =$' +f' {expected_value_model} Euro'
    l2 = r'$f(x^{{({index})}})$ = {waterfall_value} Euro'.format(index=index,
    waterfall_value=waterfall[-1])
    ax.text(expected_value_model, -1.2, l1 , va='center', ha='center')
    ax.text(waterfall[-1], 2.8, l2, va='center', ha='center')

    ax.set_yticks(range(len(labels)))
    ax.set_yticklabels(labels)
    ax.set_xlabel('Euro')

    ax.axvline(x=expected_value_model, color='grey', linestyle='--', linewidth=1)
    ax.axvline(x=waterfall[-1], color='grey', linestyle='--', linewidth=1)

    ax.set_xlim(min(waterfall)-5, max(waterfall)+5)

    for spine in ['top', 'right', 'left']:
        ax.spines[spine].set_visible(False)

    plt.tight_layout()
    plt.tick_params(left = False)
    plt.savefig(f'images/{name}', dpi=300)

"""
Create charts for obersavtion x^{(1)} and x^{(2)}
"""
# General Model Values
observation_labels = ['Groesse', 'Anzahl Zimmer', 'Entfernung zum Zentrum']
expected_value_model = 680

# Observation values and contributions for x^{(1)}
index = 1
observation_values_x1 = [100, 3, 5]
contributions_x1 = [-125, -10, 5]

plot_waterfall(expected_value_model=expected_value_model,
               observation_values=observation_values_x1,
               observation_labels=observation_labels,
               contributions=contributions_x1,
               name="model-output-x1.png", index=index)

# Observation values and contributions for x^{(2)}
index = 2
observation_values_x2 = [150, 4, 10]
contributions_x2 = [125, 10, -5]

plot_waterfall(expected_value_model=expected_value_model,
               observation_values=observation_values_x2,
               observation_labels=observation_labels,
               contributions=contributions_x2,
               name="model-output-x2.png", index=index)
```

A.3. linreg.py

```
import requests
import zipfile
from io import BytesIO
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import shap
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np
from sklearn.inspection import permutation_importance

matplotlib.use('Agg')
plt.rcParams.update({'font.size': 15})

def load_data() -> pd.DataFrame:
    """
    Loads and returns the dataset from the given URL as a Pandas DataFrame.

    Returns:
        pd.DataFrame: The loaded dataset.
    """
    url = "https://archive.ics.uci.edu/static/public/165/concrete+compressive+
    strength.zip"

    r = requests.get(url)

    if r.ok:
        with zipfile.ZipFile(BytesIO(r.content)) as thezip:
            with thezip.open("Concrete_Data.xls") as thefile:
                return pd.read_excel(thefile, header=0)
    else:
        raise Exception("Something went wrong.")

def model(X: pd.DataFrame, y: pd.Series) -> (LinearRegression, pd.DataFrame):
    """
    Fits a Linear Regression model to the given data.

    Args:
        X (pd.DataFrame): The feature matrix.
        y (pd.Series): The target variable.

    Returns:
        LinearRegression: The fitted Linear Regression model.
        DataFrame: The coefficients as DataFrame.
    """
    model = LinearRegression()
    model.fit(X, y)

    cdf = pd.DataFrame(model.coef_.round(5), X.columns, columns=['Coefficients'])
    cdf.loc['Intercept'] = model.intercept_.round(5)

    return model, cdf

def plot_residuals(y_test: pd.Series, y_pred: pd.Series) -> None:
    """
    Creates and saves residual plots.
```

```
59
60      Args:
61          y_test (pd.Series): The actual target values.
62          y_pred (pd.Series): The predicted target values.
63      """
64      residuals = y_test - y_pred
65
66      _, axs = plt.subplots(1, 2, figsize=(16, 6))
67
68      axs[0].scatter(y_test, y_pred, alpha=0.5)
69      axs[0].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--k',
         linewidth=2)
70      axs[0].set_title('Actual vs. Predicted Values')
71      axs[0].set_xlabel('Actual Values')
72      axs[0].set_ylabel('Predicted Values')
73
74      axs[1].scatter(y_pred, residuals, alpha=0.5)
75      axs[1].axhline(y=0, color='k', linestyle='--', linewidth=2)
76      axs[1].set_title('Residuals vs. Predicted Values')
77      axs[1].set_xlabel('Predicted Values')
78      axs[1].set_ylabel('Residuals')
79
80      plt.savefig('images/residuals.png', dpi=300)
81
82  def plot_corr(df: pd.DataFrame) -> None:
83      """
84      Creates and saves a correlation heatmap plot.
85
86      Args:
87          df (pd.DataFrame): The feature matrix.
88      """
89      plt.figure(figsize=(12, 6))
90      sns.heatmap(df.corr(), annot=True, cmap="YlGnBu", fmt=".2f")
91      plt.tight_layout()
92      plt.savefig('images/corr.png', dpi=300)
93
94  def plot_shap(shap_values: shap.Explanation, model: LinearRegression, X: pd.
      DataFrame, idx: int) -> None:
95      """
96      Creates and saves SHAP beeswarm, bar, and waterfall plots.
97
98      Args:
99          shap_values (shap.Explanation): SHAP values.
100         model (LinearRegression): The Linear Regression model.
101         X (pd.DataFrame): The background data for partial dependence plot.
102         idx (int): Index for the SHAP waterfall plot.
103     """
104     plt.figure(figsize=(12, 6))
105     shap.plots.beeswarm(shap_values)
106     plt.tight_layout()
107     plt.savefig('images/shap_beeswarm_plot.png', dpi=300)
108
109     plt.figure(figsize=(12, 6))
110     shap.plots.bar(shap_values)
111     plt.tight_layout()
112     plt.savefig('images/shap_bar_plot.png', dpi=300)
113
114     plt.figure(figsize=(12, 6))
115     shap.plots.waterfall(shap_values[idx])
116     plt.tight_layout()
117     plt.savefig('images/shap_waterfall_plot.png', dpi=300)
```

```
118
119     plt.figure(figsize=(12, 6))
120     shap.plots.partial_dependence("cement", model.predict, X,
121                                   model_expected_value=True,
122                                   feature_expected_value=True,
123                                   ice=False,
124                                   shap_values=shap_values[idx:idx+1,:])
125     plt.tight_layout()
126     plt.savefig('images/shap_dependence_plot.png', dpi=300)
127
128     plt.figure(figsize=(12, 6))
129     shap.plots.scatter(shap_values[:, "cement"])
130     plt.tight_layout()
131     plt.savefig('images/shap_scatter_plot.png', dpi=300)
132
133 def plot_dist(df: pd.DataFrame) -> None:
134     """
135     Creates and saves distribution and histogram plot over all
136     features of df.
137
138     Args:
139         df (pd.DataFrame): The feature matrix.
140     """
141     _, axes = plt.subplots(3, 3, figsize=(15, 12))
142     axes = axes.flatten()
143
144     for i, var in enumerate(df.columns):
145         sns.histplot(df[var], ax=axes[i], kde=True)
146         axes[i].set_title(var)
147
148     plt.tight_layout()
149     plt.savefig('images/dist.png', dpi=300)
150     plt.show()
151
152 def plot_coef(coef: pd.DataFrame) -> None:
153     """
154     Plot the coefficients of a linear model as a horizontal bar chart.
155
156     Args:
157         coef (pd.DataFrame): A DataFrame containing the model's coefficients,
158                              with feature names as the index and a 'Coefficients'
    column.
159     """
160     coef = coef.drop('Intercept', errors='ignore')
161     coef = coef.sort_values(by='Coefficients', ascending=True)
162     plt.figure(figsize=(12, 6))
163     plt.barh(coef.index, coef['Coefficients'])
164     plt.axvline(0)
165     plt.xlabel('Coefficients')
166     plt.ylabel('Features')
167     plt.savefig('images/coef.png', dpi=300)
168     plt.show()
169
170 def plot_box(df: pd.DataFrame, df2: pd.DataFrame) -> None:
171     """
172     Plot the distribution of two dataframes as a box plot.
173
174     Args:
175         df (pd.DataFrame): First dataframe.
176         df2 (pd.Dataframe): Second dataframe.
177     """
```

```
178     fig, axs = plt.subplots(1, 2, figsize=(12, 4))
179 
180     axs[0].boxplot(df2.values, vert=False)
181     axs[0].set_title('original')
182     axs[0].set_yticklabels(df2.columns)
183 
184     axs[1].boxplot(df.values, vert=False)
185     axs[1].set_title('log-transformed')
186     axs[1].set_yticklabels(df.columns)
187 
188     plt.tight_layout()
189     plt.savefig('images/boxplot.png', dpi=300)
190     plt.show()
191 
192 def plot_permutation_importance(perm_importance_train: permutation_importance,
193                                 perm_importance_test: permutation_importance,
194                                 columns: np.ndarray) -> None:
195     """
196     Plots the permutation importance of features for both training and test
        datasets.
197 
198     Args:
199         perm_importance_train (object): A fitted PermutationImportance instance
        for the training dataset.
200                                     It contains importances_mean attribute,
        representing the decrease in
201                                     model score when a feature is randomly
        shuffled.
202         perm_importance_test (object): A fitted PermutationImportance instance for
         the test dataset,
203                                    similar to perm_importance_train.
204         columns (list or array): An array or list of feature names corresponding
        to the indices in
205                                 the perm_importance objects.
206     """
207     fig, axs = plt.subplots(1, 2, figsize=(12, 4))
208 
209     sorted_idx_train = perm_importance_train.importances_mean.argsort()
210     axs[0].barh(range(len(sorted_idx_train)), perm_importance_train.
        importances_mean[sorted_idx_train], align='center')
211     axs[0].set_title('train')
212     axs[0].set_xlabel('Change in MSE')
213     axs[0].set_yticks(range(len(sorted_idx_train)), np.array(columns)[
        sorted_idx_train])
214 
215     sorted_idx_test = perm_importance_test.importances_mean.argsort()
216     axs[1].barh(range(len(sorted_idx_test)), perm_importance_test.importances_mean
        [sorted_idx_test], align='center')
217     axs[1].set_title('test')
218     axs[1].set_xlabel('Change in MSE')
219     axs[1].set_yticks(range(len(sorted_idx_test)), np.array(columns)[
        sorted_idx_test])
220 
221     plt.tight_layout()
222     plt.savefig('images/permutation_importance.png', dpi=300)
223     plt.show()
224 
225 def plot_feature_effects(coef: pd.DataFrame, X: pd.DataFrame, idx: int) -> None:
226     """
227     Create and save a box plot of feature effects.
228 
```

```
    Args:
        coef (pd.DataFrame): A DataFrame containing the model's coefficients,
                             with feature names as the index and a 'Coefficients'
    column.
        X (pd.DataFrame): The feature matrix.
        idx (int): Index for the data point to plot.
    """
    coef = coef.drop('Intercept', errors='ignore')
    feature_effects = X * coef['Coefficients']
    feature_names = X.columns

    plt.figure(figsize=(12, 6))
    plt.boxplot(feature_effects.values, vert=False)
    plt.plot(X.iloc[idx] * coef['Coefficients'], range(1, len(X.iloc[idx]) + 1), '
    rx', markersize=10)
    plt.axvline(0, linestyle='--', color='black')
    plt.xlabel('Feature Effects')
    plt.ylabel('Features')
    plt.yticks(range(1, len(feature_names) + 1), feature_names)
    plt.tight_layout()
    plt.savefig('images/feature_effects_boxplot.png', dpi=300)
    plt.show()

def calculate_shap_contribution(coef: pd.DataFrame, shap_values: shap.Explanation,
     feature: str, idx: int):
    """
    Calculate the average contribution of the feature to the model prediction.

    Args:
        coef: A pandas Dataframe containing the coefficients of the linear
    regression model.
        shap_values: A SHAP Explanation object containing the SHAP values for the
    model features.
        feature: The name of the feature.
        idx: The index of obersavtion to calculate the value for.

    Returns:
    - The average SHAP contribution for the feature.
    """
    coef = coef.loc[feature].values[0]
    feature_values = shap_values[:, feature].data
    mean = np.mean(feature_values)
    shap_contribution = coef * (feature_values[idx] - mean)
    return shap_contribution


df = load_data()

df = df.rename(
    columns={
        'Cement (component 1)(kg in a m^3 mixture)':'cement',
        'Blast Furnace Slag (component 2)(kg in a m^3 mixture)':'blast',
        'Fly Ash (component 3)(kg in a m^3 mixture)':'ash',
        'Water  (component 4)(kg in a m^3 mixture)':'water',
        'Superplasticizer (component 5)(kg in a m^3 mixture)':'superplasticizer',
        'Coarse Aggregate  (component 6)(kg in a m^3 mixture)':'coarse',
        'Fine Aggregate (component 7)(kg in a m^3 mixture)':'fine',
        'Age (day)':'age',
        'Concrete compressive strength(MPa, megapascals) ': 'strength'
    }
)
```

```
285 df = df.drop_duplicates()
286 df_original = df.copy()
287
288 print(df.head())
289 print(df.isnull().sum())
290 print(df.describe().T)
291
292 for column in df.columns:
293     df[column] += 1
294     df[column] = np.log(df[column])
295
296 plot_dist(df=df)
297 plot_box(df=df, df2=df_original)
298
299 X = df.drop(['strength'], axis=1)
300 y = df['strength']
301
302 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
        random_state=42)
303
304 linreg, coef = model(X=X_train, y=y_train)
305 y_pred = linreg.predict(X_test)
306
307 perm_importance_train = permutation_importance(linreg, X_train, y_train)
308 perm_importance_test = permutation_importance(linreg, X_test, y_test)
309 plot_permutation_importance(perm_importance_train, perm_importance_test, X.columns
        )
310
311 explainer = shap.Explainer(linreg.predict, X)
312 shap_values = explainer(X_test)
313
314 mae = mean_absolute_error(y_test, y_pred)
315 mse = mean_squared_error(y_test, y_pred)
316 train_score = linreg.score(X_train, y_train)
317 test_score = linreg.score(X_test, y_test)
318
319 print(f"Mean Absolute Error (MAE): {mae:.2f}")
320 print(f"Mean Squared Error (MSE): {mse:.2f}")
321 print(f"Root Mean Squared Error (RMSE): {mse ** 0.5:.2f}")
322 print(f"Training Score (R^2): {train_score:.4f}")
323 print(f"Test Score (R^2): {test_score:.4f}")
324
325 plot_coef(coef=coef)
326 print(coef)
327 plot_corr(df=df)
328 plot_residuals(y_test=y_test, y_pred=y_pred)
329
330 plot_shap(shap_values=shap_values, model=linreg, X=X_test, idx=0)
331 plot_feature_effects(coef=coef, X=X_test, idx=0)
332
333 print(calculate_shap_contribution(coef=coef, shap_values=shap_values, feature="
        cement", idx=0))
```