# Shapley-Werte zur Erklärung von Finanzmodellen

Daniela Söllheim

Geboren am 05. Oktober 2000 in Bonn

21. Juli 2022

Bachelorarbeit Mathematik

Betreuer: Prof. Dr. Jochen Garcke

Zweitgutachter: Dr. Daniel Oeltz

Institut für Numerische Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

# Danksagung

Gerne möchte ich diese Stelle nutzen, um mich bei denen zu bedanken, die mich während meines gesamten Studiums und insbesondere in der Phase meiner Bachelorarbeit unterstützt haben. Zu Beginn danke ich Professor Dr. Jochen Garcke für die Möglichkeit, meine Bachelorarbeit anwendungsbezogen in einem sehr spannenden und aktuellen Bereich der Numerik schreiben zu dürfen. Ergänzend dazu gilt mein Dank Dr. Daniel Oeltz für die Übernahme der Zweitkorrektur und vielen anregenden Diskussionen über das Thema dieser Arbeit.

Besonders möchte ich mich bei Johanna Albrechts bedanken, die mir mit viel Engagement und noch mehr Geduld bei kleineren und größeren Problemen in Verständnis- und Programmierfragen stets zur Seite gestanden hat. Ein herzliches „Dankeschön!“ gilt auch meinen Freunden Carlotta, Judith, Caro, Sharan, Moritz und Martin für ihre Zeit und Mühe sowie konstruktive Kritik und motivierenden Worte und Ben für seine Mithilfe an der zeitgerechten Fertigstellung der rechenaufwendigen Experimente durch Bereitstellung seiner Rechenkapazitäten.

Abschließend danke ich meiner Familie und meinem Freund für die bedingungslose Unterstützung, die mir in vielen Situationen während des gesamten Studiums das Leben erleichtert hat.

# Inhaltsverzeichnis

# 1. Einleitung

*Wie kommt der Preis zustande?*
*Weshalb habe ich ein erhöhtes Krebsrisiko?*
*Wieso wird mir dieser Vertrag empfohlen?*

Eine wichtige Aufgabe, die durch Modelle des maschinellen Lernens übernommen werden kann, ist die Vorhersage von Werten auf der Basis zuvor betrachteter Daten. In etlichen Alltagsbereichen findet dies Anwendung: In der Medizin, z. B. bei der Vorhersage von Hypoxämierisiken (Sauerstoffmangel) [Lun+18], bei der Aufdeckung und frühzeitigen Erkennung von Kriminalität wie Betrug im Finanzsektor und Geldwäsche [Sud+10] oder bei Kreditwürdigkeitsprüfungen, z. B. dem Ausfall von (Hypotheken-)Krediten [Kva+18].
Dabei können die erzeugten Prognosen auf den Nutzer durchaus willkürlich wirken. In der Folge stellt sich die Frage, ob man dem Modell überhaupt vertrauen kann.

Um Vorhersagen eines Modells für Nutzer sowie Entwickler verständlicher zu machen, wurden neue Methoden – die sogenannten Erklärungsmodelle – entwickelt, die komplexe Modelle erklären sollen und damit ihre Vertrauenswürdigkeit steigern können. Wir verstehen diese als interpretierbare Approximationen des komplexeren Originalmodells.

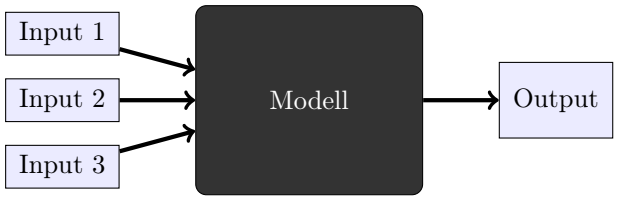



Abbildung 1.1.: **Blackbox-Modell.** Das Modell verarbeitet die Inputs auf der linken Seite zu einem Output (hier: einer Vorhersage) auf der rechten Seite.

**Beispiel 1.0.1.** *Wir möchten den Preis unserer Immobilie schätzen. Heutzutage ist das kein Problem mehr – schnell, kostenfrei, aktuell, unverbindlich und außerdem noch hochwertig: Etliche Seiten im Internet bieten uns die entsprechenden Möglichkeiten. Man wählt für eine Wohnungspreisschätzung einfach die richtigen Merkmale aus.*

*Nach wenigen Sekunden wird eine Zahl präsentiert – wir kennen jetzt einen ungefähren Verkaufspreis unserer Wohnung.*

*Aber wie kommt dieser Preis zustande?*
*Welche Parameter haben welchen Einfluss auf die Preisvorhersage?*

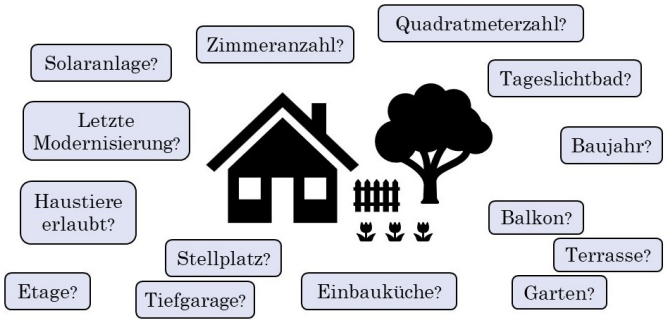



Abbildung 1.2.: **Wohnungspreisparameter.** Einige Parameter, die bei der Wohnungspreisschätzung angegeben werden und den Preis beeinflussen können.

In dieser Arbeit wird eine Methode vorgestellt, mit der man genau solche Fragen beantworten kann.

Die Wohnungspreisschätzung entspricht einem Blackbox-Modell wie in Abbildung 1.1. Als Inputs sehen wir einige Merkmale, die in Abbildung 1.2 aufgeführt sind. Der Output ist die Vorhersage des Wohnungspreises.

In den Erklärungsmodellen wird zwischen lokalen und globalen Erklärungen unterschieden. Globale Erklärungen beziehen sich auf das gesamte Modell und sind daher vor allem in der Entwicklungsphase für Fachleute relevant. Lokale Erklärungen hingegen können auch für den Endnutzer interessant sein, da sie Informationen über eine Vorhersage für eine einzelne Beobachtung zu einem bestimmten Zeitpunkt liefern [ZRS22]. In dieser Arbeit beziehen wir uns auf lokale Erklärungen, die modellunabhängig sind. Für eine Vorhersage eines unbestimmten Modells finden diese eine interpretierbare Erläuterung.
In jener Sparte gibt es wiederum verschiedene Ansätze. Wir untersuchen folgend ein Erklärungsmodell, das die Shapley-Werte miteinbezieht, die wünschenswerte Eigenschaften aufweisen, welche im zweiten Kapitel genauer betrachtet werden. Shapley-Werte wurden 1953 vom Mathematiker und Wirtschaftswissenschaftler Lloyd Shapley entwickelt, um eine faire Verteilung der einzelnen Auszahlungen an einen jeden Spieler unter komplizierten Beteiligungs- und Kostenfunktionen in einer Koalition – einem Team – zu ermitteln.

Die Idee dahinter kann gut auf Erklärungsmodelle übertragen werden: In einem Erklärungsmodell soll jedem Merkmal ein Beitrag auf Grundlage einer komplizierten Funktion zugewiesen werden. Die Merkmale werden also als Spieler und die Vorhersage als Gesamtauszahlung interpretiert, sodass die Differenz zwischen der Durchschnittsvorhersage, also beispielsweise der durchschnittlichen Vorhersage aller Wohnungen,

und der Betrachteten, der einen Wohnung, für die wir den Preis geschätzt haben, genau der Summe der Beiträge der einzelnen Merkmale entspricht.

Dabei ist die konkrete Berechnung der Shapley-Werte sehr aufwendig, da sie exponentiell mit der Anzahl der verwendeten Merkmale wächst. Um den Rechenaufwand zu reduzieren, wurden verschiedene Approximationsvarianten entwickelt, von denen wir die Kernel-SHAP-Methode genauer betrachten werden. Sie bringt allerdings neben vieler wünschenswerter Eigenschaften ein Problem mit sich: Sie verwendet die Annahme, dass alle Merkmale unabhängig sind. Das kann zu weniger realistischen und aussagekräftigen Ergebnissen führen, da reale Datensätze häufig gewisse Korrelationen aufweisen.

Im wesentlichen Kern dieser Bachelorarbeit werden wir die Kernel-SHAP-Methode dahingehend erweitern, dass auch Abhängigkeiten zwischen den Merkmalen in Datensätzen bei der Erklärung von Modellen berücksichtigt werden. Um diese neue Implementierung zu validieren, nutzen wir einen künstlich generierten Datensatz und vergleichen die Ergebnisse unserer Modifizierungen mit der originalen Kernel-SHAP-Methode und den exakten Shapley-Werten. Zudem werden wir die Methoden auf ein Kreditmodell anwenden und die Ergebnisse auswerten.

## Aufbau der Arbeit

In den Kapiteln 2 und 3 werden Einordnung und Berechnung der Shapley-Werte und ihre Verbindung zur SHAP-Methode vorgestellt. Die Kernel-SHAP-Methode sowie der dazugehörige Algorithmus werden in Kapitel 4 präsentiert. Durch drei verschiedene Ansätze der Berücksichtigung von Abhängigkeitsstrukturen im zugrundeliegenden Datensatz wird Kernel SHAP in Kapitel 5 erweitert. Die erste Modifikation basiert auf der Idee, dass die Daten multivariat normalverteilt sind, die zweite Idee beinhaltet die Verwendung einer Gauß-Copula zur Modellierung der zugrundeliegenden Abhängigkeitsstruktur und die dritte Modifikation ähnelt einer Kerndichtenschätzung, die von Aas et al. [AJL21] entwickelt wurde. Ergänzend dazu gibt es zwei zusätzliche Erweiterungsansätze, die eine Mischung aus der ersten und dritten bzw. zweiten und dritten Modifikation sind. In Kapitel 6 prüfen wir die Auswirkungen der Erweiterung zum einen anhand eines künstlichen generierten Datensatzes zur Validierung und zum anderen anhand eines Datensatzes zum Kreditausfall. Es zeigt sich, dass die unterschiedliche Behandlung der Abhängigkeit zwischen den Variablen zu unterschiedlichen Shapley-Werten führt. Kapitel 7 besteht aus einer kurzen Zusammenfassung der Ergebnisse sowie einem Fazit.

## Eigener Beitrag

- Detaillierte Aufbereitung der Shapley-Werte im Rahmen der koalitiven Spieltheorie nach Shapley [Sha53] sowie Darstellung der Fundamente von Erklärungsmodellen und der Übertragung von Shapley-Werten auf diese

- Präsentation der SHAP-Methode sowie Vorstellung und Einordnung der Unterkategorie Kernel SHAP, wobei neben der mathematischen Sicht auch das Vorgehen des Algorithmus eine wesentliche Rolle spielt
- Eigenständige, gründliche Einarbeitung in das bestehende SHAP-Paket in Python [Lun22]
- Redigierte Darstellung der vier im Hauptpaper [AJL21] entwickelten Ansätze zwecks Modifizierung der Kernel-SHAP-Methode, wieder mit besonderem Hinblick auf Theorie und Programmierung
- Eigenständige Implementierung der vier Ansätze in den bereits bestehenden Rahmen des SHAP-Paketes in Python
- Empirische Validierung der Performance der neuen Ansätze auf Grundlage künstlich generierter Daten in verschiedenen Dimensionen
- Anwendung des Verfahrens auf Daten eines Finanzmodells zur Ausfallschätzung von Krediten und Interpretation der Ergebnisse dieses Experiments

# 2. Shapley-Werte

In diesem Abschnitt werden wir die Shapley-Werte nach Lloyd Shapleys Paper [Sha53] formal einführen und ihre Berechnung anhand eines Beispiels verdeutlichen. Dazu widmen wir uns unter anderem vier wichtigen Eigenschaften, welche die Shapley-Werte als Zahlungsregel hervorheben.

Shapley-Werte wurden in der koalitiven Spieltheorie entwickelt, um die von einer Koalition erzielten Auszahlung fair an die Spieler aufzuteilen. ‚Fair' bedeutet in diesem Kontext ihrem Beitrag zur Auszahlung entsprechend. Dieser kann ein Gewinn oder aber auch zu begleichende Kosten sein, die entstanden sind.

## 2.1. Berechnung von Shapley-Werten

**Definition 2.1.1** (**Shapley-Wert**)**.** Bei Betrachtung eines Spiels mit $M$ Spielern, Koalitionen $S \subseteq \mathcal{M} = \{1, ..., M\}$ mit $|S|$ Spielern und einer Beitragsfunktion $\nu(S)$, die jeder Koalition $S$ einen Wert zuordnet, kann dieser Wert mithilfe des Shapley-Werts fair an die Spieler verteilt werden. Der Anteil, den Spieler $j$ verdient, ist:

$$\phi_j(\nu) = \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \underbrace{\left(\nu(S \cup \{j\}) - \nu(S)\right)}_{\text{Marginale Beiträge}}. \tag{2.1.1}$$

Wenn wir die Formel genauer betrachten, können wir feststellen, dass über die Teilmengen $\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}$ summiert wird. Anders ausgedrückt summiert man also die marginalen Beiträge gewichtet auf, die der Spieler $j$ in den möglichen Koalitionen erzielt. Die marginalen Beiträge eines Spielers entsprechen der Differenz der Auszahlung, welche die Koalition mit dem $j$-ten Spieler erzielt, und der Auszahlung, welche diese ohne den $j$-ten Spieler erzielt. Wenn aus dem Kontext erschlossen werden kann, welche Beitragsfunktion $\nu$ verwendet wird, schreiben wir zwecks Vereinfachung $\phi_j$ anstelle von $\phi_j(\nu)$.

Um das Ganze zu veranschaulichen, betrachten wir im Folgenden ein Beispiel.

**Beispiel 2.1.2.** *Ein Team aus drei Spielern A, B und C erreicht in einer Quizshow einen gemeinsamen Gewinn von 100 €. Nun stellt sich die Frage, wie das Preisgeld aufgeteilt werden soll.*

*Sie stimmen in der Meinung überein, dass es anhand ihres Beitrages in der Show verteilt werden muss.*

*Nur sind diese sehr unterschiedlich verteilt. Spieler A bedient sehr viele Themengebiete, Spieler B hingegen ist nur mit einem einzigen vertraut. Der dritte Spieler C hat eine ausgezeichnete Reaktionszeit, kann dafür aber inhaltlich wenig beitragen. Um eine ‚faire' Verteilung zu erreichen, berechnen sie die einzelnen Shapley-Werte. Zuerst bestimmen sie den Gewinn, den die einzelnen Koalitionen erzielen können:*

| No. | Koalition | Auszahlung | No. | Koalition | Auszahlung |
|---|---|---|---|---|---|
| 1 | | $\to 0$ | 5 | A B | $\to 95$ |
| 2 | A | $\to 60$ | 6 | A C | $\to 90$ |
| 3 | B | $\to 5$ | 7 | B C | $\to 10$ |
| 4 | C | $\to 0$ | 8 | A B C | $\to 100$ |

Abbildung 2.1.: **Auszahlung bei verschiedenen Koalitionen.** In jeder Spalte ist rechts die Auszahlung zu der Koalition in der Mitte zu finden.

*Insgesamt gibt es acht verschiedene Koalitionen.*
*Für jeden Spieler kann jetzt die Formel für den Shapley-Wert aufgestellt werden:*

$$\begin{aligned}\phi_A = &\frac{0!\cdot 2!}{3!}(60-0)+ \frac{1!\cdot 1!}{3!}(95-5)+ \frac{1!\cdot 1!}{3!}(90-0)+ \frac{2!\cdot 0!}{3!}(100-10)\\ = &\quad 80\end{aligned}$$

$$\begin{aligned}\phi_B = &\frac{0!\cdot 2!}{3!}(5-0)+ \frac{1!\cdot 1!}{3!}(95-60)+ \frac{1!\cdot 1!}{3!}(10-0)+ \frac{2!\cdot 0!}{3!}(100-90)\\ = &\quad 12.5\end{aligned}$$

$$\begin{aligned}\phi_C = &\frac{0!\cdot 2!}{3!}(0-0)+ \frac{1!\cdot 1!}{3!}(90-60)+ \frac{1!\cdot 1!}{3!}(10-5)+ \frac{2!\cdot 0!}{3!}(100-95)\\ = &\quad 7.5\end{aligned}$$

*Der Spieler A leistet also einen Beitrag, der mit* 80 € *entlohnt werden sollte, die Entlohnung von Spieler B sollte* 12,50 € *und die von C* 7,50 € *betragen.*

Abbildung 2.2 zeigt ein Säulendiagramm, das die Gewichte

$$\frac{|S|!(M-|S|-1)!}{M!}$$

der einzelnen marginalen Beiträge in der Formel der Shapley-Werte abhängig von Koalitionsgröße darstellt.

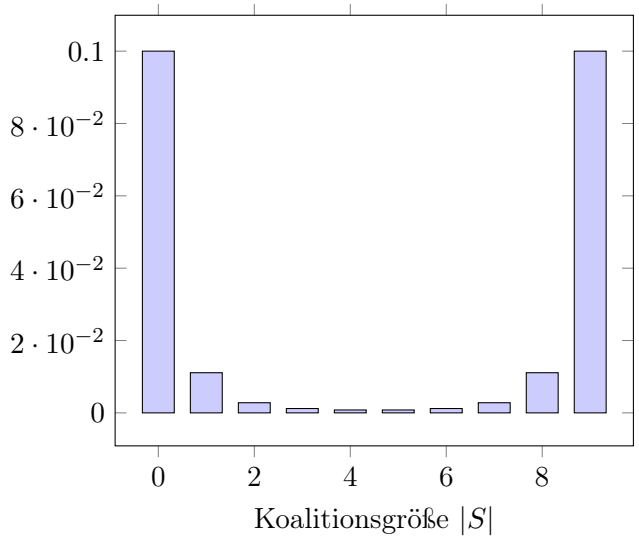



Abbildung 2.2.: **Gewichtung der marginalen Beiträge.** Für eine feste Anzahl von insgesamt zehn Spielern sieht man hier die Gewichtung der Koalitionsgrößen.

Es wird ersichtlich, dass kleine und große Koalitionen stärker gewichtet sind als diejenigen mittlerer Größe. Die Intuition dahinter besteht darin, dass es zum einen mehr Koalitionen mittlerer Größe gibt als kleine und große Koalitionen und dass zum anderen die Effekte, die einzelne Spieler haben, in kleinen Koalitionen isoliert beobachtet werden können. Ebenso bedeutet eine größere Änderung der Auszahlung in einer großen Koalition einen stärkeren Beitrag des einzelnen Spielers.

## 2.2. Eigenschaften von Shapley-Werten

Shapley-Werte besitzen vier nennenswerte Eigenschaften [ŠK10]:

1. **Effizienz.** Die Summe aller Shapley-Werte entspricht der Auszahlung der Koalition, an der alle beteiligt sind:

$$\sum_{j=0}^{M} \phi_j = \nu(\mathcal{M}).$$

2. **Symmetrie.** Die Shapley-Werte zweier Spieler, die zu jeder möglichen Koalition den gleichen Beitrag erbringen, müssen gleich sein. Seien $i \neq j$ zwei Spieler. Wenn

$$\nu(\mathcal{S} \cup \{i\}) = \nu(\mathcal{S} \cup \{j\}) \text{ für alle } \mathcal{S} \subset \mathcal{M} \text{ mit } i, j \notin \mathcal{S},$$

$$\text{dann gilt } \phi_i = \phi_j.$$

3. **Null-Spieler.** Ein Spieler $i$, der in keiner Koalition einen Beitrag leistet, muss einen Shapley-Wert von null haben. Wenn

$$\nu(\mathcal{S} \cup \{i\}) = \nu(\mathcal{S}) \text{ für alle } \mathcal{S} \subset \mathcal{M},$$

$$\text{dann gilt } \phi_i = 0.$$

4. **Linearität.** Für zwei verschiedene Spiele mit unterschiedlichen Verteilungsfunktionen $\nu$ und $\omega$ und einer reellen Zahl $a$ gilt:

$$\phi_i(\nu + \omega) = \phi_i(\nu) + \phi_i(\omega) \text{ für alle } i \text{ und}$$

$$\phi_i(a\nu) = a\phi_i(\nu),$$

wobei $(\nu + \omega)(\mathcal{S}) := \nu(\mathcal{S}) + \omega(\mathcal{S})$ und $(a\nu)(\mathcal{S}) := a \cdot \nu(\mathcal{S})$.

Tatsächlich ist Auszahlung der Shapley-Werte entsprechend die einzige Zahlungsregel, die diese Eigenschaften erfüllt. Weitere Informationen sowie der Beweis sind nachzulesen in [Sha53] und [Che+20].

# 3. Anwendung in Erklärungsmodellen

In diesem Kapitel wollen wir Shapley-Werte auf Erklärungsmodelle übertragen. Bevor wir uns dem eigentlichen Thema widmen, geben wir eine kurze Einführung in benötigte Grundkenntnisse des maschinellen Lernens, bei der wir uns zur Vereinfachung auf Modelle der Form $f : \mathbb{R}^M \to \mathbb{R}$ beschränken. Allgemein kann man das nachstehend beschriebene Prinzip ebenfalls für das Erklären von Bildern oder Texten erweitern.

Anschließend betrachten wir additive ‚Feature-Attribution'-Modelle (additive Merkmalszuschreibungsmodelle) und lernen erstmals SHAP-Werte und ihre Funktionsweise nach Lundberg und Lee kennen [LL17].

## 3.1. Grundkenntnisse und Notation

Zusammengefasst ist das Vorgehen bei überwachtem Lernen, dem Sektor des maschinellen Lernens, den wir betrachten, das Folgende: Zu einem Datensatz $X$ kennt man die dazugehörigen Ergebnisse $Y$. Im Beispiel 1.0.1 bestände der Datensatz $X$ aus Instanzen, die je eine Wohnung repräsentieren, gefüllt mit den jeweiligen Merkmalen (Quadratmeterzahl, Zimmeranzahl, usw.), für die der Wohnungspreis bereits bekannt ist. Diese Preise zu den jeweiligen Zeilen aus $X$ bilden unsere Ergebnisse $Y$ (auch ‚targets', zu Deutsch: Ziele). Auf Grundlage derer trainiert man einen Algorithmus, beispielsweise Modelle wie eine lineare Regression oder Entscheidungsbäume. Das bedeutet, dass dem Modell die Daten $X$ und $Y$ zur Verfügung gestellt werden und es lernen soll, für einen Input $x$ einen Output $f(x)$ zu erzeugen, der möglichst dem Ergebnis $y$ entspricht.

**Beispiel 3.1.1** (Lineare Regression)**.** *Für eine Instanz $x$ mit $n$ Merkmalen ergibt sich ein lineares Regressions-Modell der Form*

$$f(x) = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n.$$

*Die Parameter $\beta_0, ..., \beta_n$ sind die gelernten Koeffizienten des Modells, $\varepsilon = y - f(x)$ bezeichnet einen Störfaktor, welcher die Differenz zwischen der Modellvorhersage $f(x)$ und dem tatsächlichen Ergebnis $y$ beschreibt.*

Vorhersagen, die aus solchen Modellen erzeugt werden, möchten wir nun im Folgenden mithilfe der Shapley-Werte erklären, indem wir sie auf die Beiträge der einzelnen Merkmale zurückführen.

Als ein bedeutendes Problem wird sich dabei herausstellen, dass wir zur Berechnung dieser – wie auch in dem Beispiel 2.1.2 – Teilmengen von Merkmalen betrachten müssen. Im Allgemeinen können Modelle des maschinellen Lernens allerdings nicht mit dem Weglassen von Merkmalen umgehen, auch weil es keine Möglichkeit gibt, das Fehlen eines Merkmals darzustellen: Repräsentiert ein Merkmal beispielsweise die Temperatur, bedeutet der Wert null ‚0°C' und nicht die Abwesenheit eines Merkmals. Nachfolgend wird ein fehlendes Merkmal deshalb durch einen Stern ‚$\star$' dargestellt.

Ergänzend dazu führen wir für eine feste Instanz $x$ die Menge $Z_x$ ein. Sie ist definiert durch

$$Z_x := \{z_{\mathcal{S}} \mid \mathcal{S} \subseteq \mathcal{M} = \{1, ..., M\}\},$$

wobei die Elemente $z_{\mathcal{S}}$ Vektoren der Form

$$(z_{\mathcal{S}})_i = \begin{cases} x_i, & i \in \mathcal{S} \text{ , d. h. wir verwenden das } i\text{-te Merkmal} \\ \star, & \text{sonst} \end{cases}$$

sind.

Für eine gegebene Funktion

$$f_x : \{0,1\}^M \to \mathbb{R}$$

definieren wir die zwei nachstehenden Hilfsfunktionen

$$h_x : \{0,1\}^M \to Z_x, \qquad z'_{\mathcal{S}} \to z_{\mathcal{S}} \text{ und}$$

$$\hat{f}_x : Z_x \to \mathbb{R}, \qquad z_{\mathcal{S}} \to \hat{f}(z_{\mathcal{S}}),$$

sodass

$$\hat{f}_x(z_{\mathcal{S}}) = \hat{f}_x(h_x(z'_{\mathcal{S}})) = f_x(z'_{\mathcal{S}}) : \{0,1\}^M \to \mathbb{R}.$$

Dabei soll $f_x$ eine künstliche Erweiterung des Modells $f$ sein, das anders als das Originalmodell mit fehlenden Merkmalen arbeiten kann. Wichtig ist, dass diese lokal für $x$ definiert ist. Wie genau wir $f_x$ wählen können, wird nachstehend in Abschnitt 3.3 erklärt.

Für einen besseren Überblick über die verschiedenen Vektoren schauen wir uns das Beispiel 2.1.2 an:

**Beispiel 3.1.2.** *Nehmen wir an, wir möchten die Vorhersage $f(x)$ des Inputs*

$$x = (5, -1, 32, 17)$$

*eines Modells $f$ erklären. Diesen Vektor können wir nun als vereinfachten binären Vektor*

$$x' = (1, 1, 1, 1)$$

*darstellen – jedes Merkmal ist angegeben.*
*Wollen wir Merkmale weglassen und z. B. nur das erste und vierte Merkmal betrach-*

*ten – die Teilmenge $\mathcal{S} = \{1, 4\}$ – so ist der zugehörige Vektor*

$$z_{\mathcal{S}} = (5, \star, \star, 17).$$

*Zur Erinnerung: Das ‚$\star$' verkörpert an dieser Stelle einen leeren Eintrag. Die vereinfachte binäre Form dieses Vektors, die wieder angibt, welche Merkmale vorhanden sind, ist*

$$z'_{\mathcal{S}} = (1, 0, 0, 1).$$

*Wenn wir eine (künstliche) Vorhersage für den Vektor $z'_{\mathcal{S}}$ erzeugen möchten, wenden wir also $f_x$ darauf an.*

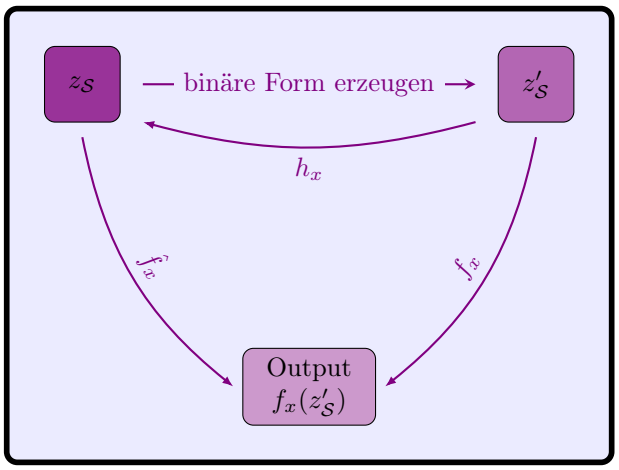



Abbildung 3.1.: **Zusammenhang der Hilfsfunktionen $h_x, \hat{f}_x$ und $f_x$, um das Auslassen von Merkmalen im Modell $f$ zu simulieren.** Eine das Beispiel 3.1.2 ergänzende strukturelle Abbildung, um den Zusammenhang zwischen den für das Weglassen der Merkmale benötigten Funktionen zu visualisieren.

## 3.2. Additive ‚Feature-Attribution'-Modelle

Die beste Erklärung für ein Modell ist natürlich dieses selbst.
Aber je komplexer das Modell wird, desto unverständlicher ist es häufig. Das verdeutlicht die Notwendigkeit von vereinfachenden Erklärungsmodellen.

Bei lokalen Erklärungsmodellen, auf die wir uns in dieser Arbeit fokussieren, bezweckt man, dass die Vorhersage des Erklärungsmodells der Vorhersage des originalen Vorhersagenmodells bei gleichen Inputs entspricht.

**Definition 3.2.1** (**Additive ‚Feature-Attribution'-Methode**). [LL17] Sei $f$ ein Vorhersagenmodell und $x$ ein einzelner Input. Additive ‚Feature-Attribution'-Me-

thoden haben ein Erklärungsmodell $g : \{0,1\}^M \to \mathbb{R}$ der Form

$$g(z'_{\mathcal{S}}) = \phi_0 + \sum_{i=1}^{M} \phi_i (z'_{\mathcal{S}})_i, \tag{3.2.1}$$

wobei $z'_{\mathcal{S}} \in \{0,1\}^M$ die vereinfachte Form des Vektors $z_{\mathcal{S}} \in Z_x$ ist, der einen Teil des Inputs $x$ darstellt, $M$ die Anzahl der Inputs und $\phi_i \in \mathbb{R}$.

*Bemerkung* 3.2.2. Da wir uns in dieser Arbeit ausschließlich auf lokale Erklärungsmodelle beziehen, gilt:

$$x' \approx z'_{\mathcal{S}} \Rightarrow f_x(x') \approx g(z'_{\mathcal{S}}).$$

Wenn also die Teilmenge betrachtet wird, bei der alle Merkmale eingeschlossen sind, soll das Ergebnis des Erklärungsmodells für diesen Datenpunkt dem Modell-Output entsprechen.

Erklärungsmodelle der Form aus Definition 3.2.1 erinnern an eine lineare Regression. Sie schreiben jedem Merkmal $i$ einen Beitrag $\phi_i$ zu. Insgesamt addieren sich die Beiträge der ‚anwesenden' Merkmale zusammen mit einer Konstanten $\phi_0$ auf die Vorhersage, die es zu erklären gilt.

### Lösung mit wünschenswerten Eigenschaften

Laut Lundberg und Lee [LL17] gibt es genau eine eindeutige Lösung für die $\phi_i$ in der Klasse der additiven ‚Feature-Attribution'-Modelle, welche die folgenden Eigenschaften erfüllt [Mol20]:

1. **Lokale Genauigkeit.** Wenn $x = h_x(x')$ gilt und alle vereinfachenden Annahmen ausgeschaltet sind, dann gilt

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i.$$

2. **Ausbleiben.** Wenn ein Merkmal nicht eingebunden ist, kann es keinen Effekt haben.

3. **Konsistenz.** Wenn die marginalen Beiträge eines Merkmals unter einem Modell in jeder Koalition größer sind als unter einem anderen, ist $\phi_i$ in dem ersten Modell ebenfalls größer als in dem zweiten.

**Satz 3.2.3.** *Es gibt genau ein Erklärungsmodell der Form aus Definition 3.2.1, das die drei vorangegangenen Eigenschaften erfüllt:*

$$\phi_i(f, x) = \sum_{z'_{\mathcal{S}} \in Z_x} \frac{|z'_{\mathcal{S}}|!(M - |z'_{\mathcal{S}}| - 1)!}{M!} [f_x(z'_{\mathcal{S}}) - f_x(z'_{\mathcal{S}} \setminus i)], \tag{3.2.2}$$

*wobei $|z'_{\mathcal{S}}|$ die Anzahl der Einträge in $z'_{\mathcal{S}}$ ist, die nicht null sind.*

Deutlich wird, dass die einzelnen Beiträge für das Erklärungsmodell der additiven ‚Feature-Attribution'-Methode genau den in Definition 2.1.1 aufgeführten Shapley-Werten entspricht. Diese $\phi_i(f, x)$ sind die Shapley-Werte des Modells.

Bereits im letzten Kapitel konnten wir feststellen, dass Modelle mit dem Weglassen von Merkmalen nicht umgehen können. Wir haben dazu $f_x$ eingeführt, was die Vorhersage für eine Teilmenge von Merkmalen erzeugen können soll. Eine Möglichkeit, diese Funktion konkret zu benennen und damit das Fehlen der Merkmale zu simulieren, findet sich in der Definition der SHAP-Werte.

## 3.3. SHAP – SHapley Additive exPlanation

SHAP-Werte steht für ‚**SH**apley **A**dditive ex**P**lanation'-Werte, was übersetzt Additive-Shapley-Erklärungswerte bedeutet. Sie wurden 2017 von Lundberg und Lee [LL17] eingeführt und basieren auf vorherigen Arbeiten von Erik Štrumbelj und Igor Kononenko [ŠK10; ŠK14]. Dabei wird in Theorem 3.2.3 beschriebenen Methode für $f_x$ der bedingte Erwartungswert $\mathbb{E}[f(x)|x_\mathcal{S} = x^\star_\mathcal{S}]$ bezüglich der bedingten Dichte $p(x_{\bar{\mathcal{S}}}|x_\mathcal{S} = x^\star_\mathcal{S})$ verwendet. Die Dichte $p$ beschreibt die Verteilung der Daten. Hierbei ist $x^\star$ ein bestimmter, zu erklärender Vektor und

$$(x_\mathcal{S})_i = \begin{cases} x^\star_i, & i\text{-tes Merkmal ist in } \mathcal{S} \text{ enthalten} \\ 0, & \text{sonst.} \end{cases}$$

$x_\mathcal{S} = x^\star_\mathcal{S}$ beschreibt also, dass die Merkmale in der Menge $\mathcal{S}$ der zu erklärenden Instanz $x^\star$ festgesetzt werden. Das Komplement von $\mathcal{S}$ wird im Folgenden mit $\bar{\mathcal{S}}$ bezeichnet. Dementsprechend gibt der Vektor $x_{\bar{\mathcal{S}}}$ den Teil des Vektors $x$ an, der nicht in $x_\mathcal{S}$ enthalten ist.

$$(x_{\bar{\mathcal{S}}})_i = \begin{cases} x_i, & i\text{-tes Merkmal ist nicht in } \mathcal{S} \text{ enthalten} \\ 0, & \text{sonst.} \end{cases}$$

Es ergibt sich

$$f_x(z'_\mathcal{S}) := \mathbb{E}[f(x)|x_\mathcal{S} = x^\star_\mathcal{S}],$$

wobei $z'_\mathcal{S}$ sich in der Wahl der Teilmenge $\mathcal{S}$ in der Bedingung des Erwartungswertes widerspiegelt. Das $\phi_i(f, x)$ in Gleichung 3.2.2 beschreibt damit die SHAP-Werte.
Als ‚Verteilungsfunktion' wird die erwartete Vorhersage für eine Instanz $x$ genommen bedingt darauf, dass gewisse Merkmale – diejenigen, die die Teilmenge $\mathcal{S}$ enthält – gegeben sind. Auf diese Weise erzeugen wir eine Vorhersage nur für die Merkmale der Menge $\mathcal{S}$. Wir lassen demnach die Merkmale, die nicht in $\mathcal{S}$ enthalten sind, weg.

Wenn die Dichte $p$ der Daten und $f_x$ bekannt sind, ist der SHAP-Wert eindeutig definiert. In der Praxis ergeben sich hier allerdings zwei Probleme: Zum einen ist $p$ in der Regel nicht bekannt und zum anderen ist die exakte Berechnung von $f_x$ sehr aufwendig, sodass sie für eine größere Anzahl an Merkmalen kaum mehr durchführbar ist. Beide Punkte tragen dazu bei, dass die exakte Berechnung der SHAP-Werte nicht

umsetzbar ist. An dieser Stelle werden verschiedene Approximationsmethoden interessant, die in unterschiedlichen Situationen angewandt werden können. Beispielsweise gibt es Linear SHAP für lineare Modelle, Deep SHAP für Neuronale Netze oder Tree SHAP für Entscheidungsbäume. Diese SHAP-Methoden sind nur auf je ein Modell anwendbar, also modell-spezifisch. Wir interessieren uns hingegen für Kernel SHAP, eine modell-agnostische Methode.

### Verschiedene Möglichkeiten für eine Lösung

Um möglichen Missverständnissen vorzubeugen, sollte an dieser Stelle die Eindeutigkeit des Erklärungsmodells in Theorem 3.2.3 genauer erläutert werden. ‚Genau ein Erklärungsmodell' bedeutet nicht, dass es nur eine mögliche Lösung gibt, die Shapley-Werte eines Erklärungsmodells zu berechnen.
Vielmehr kann für die Funktion $f_x$ eine Vielzahl verschiedener Funktionen eingesetzt werden, da nicht ganz eindeutig ist, wie ein Modell des maschinellen Lernens in einem koalitiven Spiel darzustellen ist [Che+20]. Zwar ist bei SHAP-Werten nach Lundberg und Lee [LL17] eindeutig von $f_x = \mathbb{E}[f(x)|x_{\mathcal{S}} = x^\star_{\mathcal{S}}]$ die Rede, dennoch ist dieses Prinzip ebenso mit anderen Funktionen anwendbar. Die bedingte Dichte $p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^\star_{\mathcal{S}})$, die dem Erwartungswert zugrunde liegt, wird in der Kernel-SHAP-Methode zwecks Vereinfachung durch $p(x_{\bar{\mathcal{S}}})$ ersetzt, was in Abschnitt 4.1.2 genauer ausgeführt wird. Man kann demnach durchaus überlegen, welche Funktionen dafür infrage kommen könnten. Neben dem bedingten Erwartungswert könnten beispielsweise der bedingte Mittelwert oder Pfadintegrale gewählt werden [AJL21].

Der Vorteil von $\mathbb{E}[f(x)|x_S = x^\star_S]$ liegt darin, dass zum einen die gesamte Wahrscheinlichkeitsdichte zusammengefasst und zum anderen ein üblicher Schätzer für Vorhersagenmodelle gewählt wird. Insbesondere mit Letzterem wird eine intuitiv gut nachvollziehbare Wahl getroffen. Zudem kann er Charnes et al. [Cha+88] nach als Minimierer der mittleren quadratischen Abweichungsfunktion für eine bestimmte Vorhersage $f(x^\star)$ fungieren, was im Folgenden noch sehr hilfreich sein wird.

Ein Nachteil der Kernel-SHAP-Methode in Verbindung mit der gewählten Funktion wird die Erzeugung und Betrachtung unmöglicher Datenpunkte auf Basis der oben kurz angeschnittenen Vereinfachung der Dichte sein [FRF20].

In jedem Fall bedeutet eine andere Funktion $f_x$ auch entsprechend andere Ergebnisse für die Shapley-Werte. Für verschiedene Zwecke (Datenbewertungen, globale/lokale Modelle) können unterschiedliche Funktionen sinnvoll sein [Che+20; CLL20]. Von der Wahl der richtigen Funktion kann demnach die Aussagekraft des gesamten Erklärungsmodells abhängen. Indirekt beeinflusst diese ebenfalls, ob dieses Modell als vertrauenswürdig eingestuft wird und damit auch, ob es in der Praxis verwendet wird.

## 3.4. Intuitive Übertragung der Shapley-Werte auf SHAP-Werte

Wir möchten die Situation der Shapley-Werte nun auf unser Vorhersagenmodell übertragen [Mol20]:

| **Shapley-Werte** | | **SHAP** |
|---|---|---|
| Spieler | $\Leftrightarrow$ | Merkmale |
| Auszahlung | $\Leftrightarrow$ | Einzelne Vorhersage |
| Koalition | $\Leftrightarrow$ | Teilmenge an Merkmalen |
| Beitragsfunktion $\nu(S)$ | $\Leftrightarrow$ | $f_x(z'_{\mathcal{S}})$ |

Abbildung 3.2.: **Übertragung von Shapley-Werten auf Erklärungsmodelle.** Wichtige Bestandteile im Vergleich zwischen den originalen Shapley-Werten und der Anwendung in Erklärungsmodellen.

Was bei den Shapley-Werten die Auszahlung, die auf die einzelnen Spieler zu verteilen war, entspricht, wird hier als Vorhersage, die man in die Beiträge der einzelnen Merkmale zurückführen möchte, betrachtet. Statt einer Koalition bzw. einem Team $\mathcal{S}$ liegt hier eine Teilmenge aus Merkmalen $\mathcal{S}$, die festgesetzt wurden, vor. Statt einer Beitragsfunktion $\nu(S)$, die jeder Koalition eine Auszahlung zuweist, untersuchen wir die erwartete Vorhersage eines Modells mit der Einschränkung, dass nur die Merkmale in unserer festgesetzten Teilmenge $\mathcal{S}$ gegeben sind.

## 3.5. Intuitive Funktionsweise

Anhand des nachstehenden Diagramms in Abbildung 3.3 soll die intuitive Funktionsweise der SHAP-Werte veranschaulicht werden. Diese haben alle Methoden (Kernel SHAP, Linear SHAP, usw.) gemein. Die mathematisch formale Einführung der Kernel-SHAP-Methode ist in Kapitel 4 zu finden.

Abbildung 3.3 ist einem Diagramm zur Veranschaulichung der SHAP-Werte von Lundberg und Lee [LL17] nachempfunden.

Während eine gängige – aber falsche – Interpretation der Shapley-Werte in Erklärungsmodellen besagt, dass sie die jeweiligen Beiträge an der gesamten zu erklärenden Vorhersage beschreiben, müssen wir an dieser Stelle eine Unterscheidung machen: Tatsächlich geben die Shapley-Werte in Erklärungsmodellen nur die Beiträge an, die die Differenz zwischen der erwarteten Vorhersage und der zu erklärenden Vorhersage ausmachen.

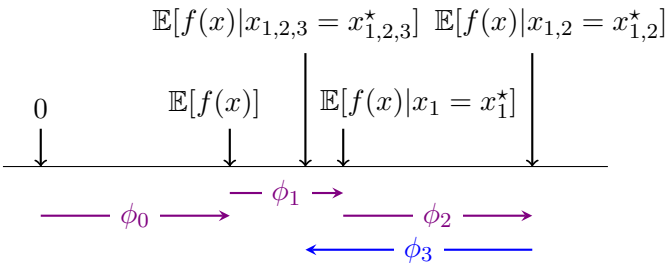



Abbildung 3.3.: **Funktionswerte der SHAP Werte.** Beginnend bei dem Erwartungswert der Vorhersage kann man dem Diagramm die einzelnen Beiträge bzw. deren Auswirkungen auf die Vorhersage entnehmen. Die horizontalen Pfeile unterhalb der Linie beschreiben die ermittelten Beiträge in einer festen Permutation, die Shapley-Werte. Die vertikalen Pfeile zur Linie hin beschreiben die Ansiedlung gewisser Vorhersagen unter unterschiedlichen Bedingungen.

Im ersten Schritt des Vorgehens verschiebt $\phi_0$ also den Startpunkt von null zur erwarteten Vorhersage. Es gilt

$$\phi_0 = \mathbb{E}[f(x)],$$

womit $\phi_0$ kein Shapley-Wert im klassischen Sinn ist.

Von dort aus beschreibt der Pfeil $\phi_1$ den Einfluss des ersten Merkmals auf die Vorhersage. Er beginnt bei der erwarteten Vorhersage ohne Berücksichtigungen und endet bei der erwarteten Vorhersage unter der Bedingung, dass das erste Merkmal festgehalten wurde. Genau gleich ist das Vorgehen bei den nächsten Merkmalen.

Besonderes Augenmerk sollte darauf gelegt werden, dass durchaus unterschiedliche Ergebnisse für die einzelnen Beiträge bei unterschiedlichen Permutationen herauskommen können. Ebenso wie in Beispiel 2.1.2, in dem verschiedene Permutationen betrachtet werden müssen, um den korrekten Beitrag zu ermitteln.

Um alle möglichen Teilmengen von Merkmalen miteinzubeziehen, muss jedes dieser Merkmale einmal inkludiert und exkludiert in jeder ‚Koalition' vorkommen. Bei einem Modell mit $M$ Merkmalen muss man also $2^M$ mögliche Teilmengen von Merkmalen untersuchen. Die Anzahl der möglichen Teilmengen wächst exponentiell mit der Anzahl der Merkmale. Bei 30 Merkmalen müssen bereits

$$2^{30} = 1.073.741.824$$

mögliche Teilmengen betrachtet werden. Je höher die Anzahl der Merkmale, desto aufwendiger und komplexer ist die exakte Berechnung.

Dieses Problem kann man vermeiden, indem geschickte Approximationsmethoden verwendet werden. Eine dieser Methoden werden wir im nächsten Kapitel genauer beleuchten.

# 4. Kernel SHAP

Anders als die meisten Verfahren der SHAP-Kategorie, wie z. B. Deep SHAP oder Tree SHAP, ist die Kernel-SHAP-Methode modell-agnostisch. Das bedeutet, dass sie auf jedes Modell des maschinellen Lernens anwendbar ist, was sie zu einer sehr beliebten Methodik macht. Auch dieses Kapitel basiert auf der von Lundberg und Lee vorgestellten Methodik [LL17] und orientiert sich an der strukturierten und detailreicheren Aufarbeitung des Hauptpapers [AJL21].

## 4.1. Bestandteile

Die Kernel-SHAP-Methode besteht aus zwei wichtige Bausteinen. Zum einen muss eine geschickte und damit auch berechenbare Approximation der Shapley-Werte eingeführt werden und zum anderen müssen wir die Beitragsfunktion $f_x$ schätzen.
Unten stehend werden wir beide Bausteine aufarbeiten, beginnend mit der Approximation unter der Annahme, dass $f_x$ bekannt ist.

### 4.1.1. Approximation der Shapley-Werte

Unter der Annahme, dass $f_x = \mathbb{E}[f(x)|x_{\mathcal{S}} = x^\star_{\mathcal{S}}]$ bekannt ist, betrachten wir eine äquivalente Darstellung der Shapley-Werte in Erklärungsmodellen.
Nach Charnes et al. kann man die Shapley-Werte ebenfalls als optimale Lösungen für die gewichtete Methode der kleinsten Quadrate (engl. weighted least squares, WLS) definieren [LL17; Cha+88]:

**Satz 4.1.1.** *Die Shapley-Werte entsprechen der optimalen Lösung des folgenden Minimierungsproblems*

$$\arg\min_g \sum_{z'_{\mathcal{S}} \subseteq Z_x} \Big[ f_x(z'_{\mathcal{S}}) - g(z'_{\mathcal{S}}) \Big]^2 \cdot \pi_{x'}(z'_{\mathcal{S}}), \tag{4.1.1}$$

*wobei $g(z'_{\mathcal{S}}) = \phi_0 + \sum_{i=1}^{M} \phi_i (z'_{\mathcal{S}})_i$ gemäß dem Erklärungsmodell aus Definition 3.2.1 gewählt wird.*

*Die Shapley-Kernel-Gewichte sind dabei definiert durch*

$$\pi_{x'}(z'_{\mathcal{S}}) = \frac{M-1}{\binom{M}{|z'_{\mathcal{S}}|} |z'_{\mathcal{S}}| (M - |z'_{\mathcal{S}}|)}. \tag{4.1.2}$$

Das zunächst auftretende Problem des Nullteilens bei $|z'_{\mathcal{S}}| \in \{0, M\}$ wird in der Praxis durch sinnvolle Optimierung vermieden. Die Lösung $\phi$ wollen wir im Folgenden genauer bestimmen.

Dafür schreiben wir das WLS Problem in einer anderen Form nieder [AJL21].
Wir verwenden die folgende Terminologie:

- eine binäre Matrix $Z$ der Form $2^M \times (M+1)$, die alle möglichen Teilmengen enthält. Dabei findet man in den einzelnen Zeilen alle möglichen Teilmengenkombinationen der $M$ Merkmale. Die erste Spalte enthält dabei nur Einsen (später das $\phi_0$) und der Eintrag $Z_{i,j}$ ist 1, wenn in der $i$-ten Koalition das $j$-te Merkmal enthalten ist, und andernfalls 0.
- eine Diagonalmatrix $W$ der Form $2^M \times 2^M$, die die Kernelgewichte $\pi_{x'}(z')$ enthält. Selbstverständlich stimmt die Reihenfolge der Teilmengen der Matrix $Z$ überein.
- einen Vektor $\phi$, der die einzelnen Einträge $\phi_0$, $\phi_1$,..., $\phi_M$ enthält.
- einen Vektor $\nu$, der die Beitragsfunktion $f_x$ darstellt.

Insgesamt kann das in 4.1.1 stehende Problem also umgeschrieben werden zu

$$(\nu - Z\phi)^T W(\nu - Z\phi). \tag{4.1.3}$$

Die Lösung dieses Minimierungsproblems kann dargestellt werden durch

$$\phi = (Z^T W Z)^{-1} Z^T W \nu. \tag{4.1.4}$$

Um das Problem der unendlichen Kernelgewichte $\pi_{x'}$ zu umgehen, setzen wir diese auf eine hohe Konstante $c = 10^6$ oder führen zusätzliche Bedingungen $\phi_0 = \nu(\emptyset)$ und $\sum_{j=0}^{M} \phi_j = \nu(\mathcal{M})$ ein [AJL21].

Die Lösung des Minimierungsproblems exakt zu berechnen, bleibt bei einer großen Anzahl an Merkmalen weiterhin aufwendig. In der Kernel-SHAP-Methode versucht man, diese Lösung nun zu approximieren.
Ein Großteil der zu betrachtenden Teilmengen hat ein sehr kleines Kernelgewicht $\pi_{x'}$. Das bedeutet, dass eine Vielzahl der Zeilen in $Z$ kaum zur Berechnung der Shapley-Werte beiträgt.

Das Vorgehen in Kernel SHAP wird daher weniger rechenintensiv gestaltet, indem nicht alle Teilmengen zur Berechnung der Shapley-Werte miteinbezogen werden. Stattdessen wird daraus eine Stichprobe $\mathcal{D} \subseteq \mathcal{M}$ nach der Verteilung der Kernel-Gewichte gezogen. Bei dieser Methode werden nur Daten dieser Stichprobe $Z_{\mathcal{D}}$ und $\nu_{\mathcal{D}}$ verwendet, also nur die Zeilen in $Z$ und $\nu$, die mit der Stichprobe $\mathcal{D}$ übereinstimmen.
Die Gewichtung dieser Stichprobe im WLS-Problem bleibt der ursprünglichen Gewichtung gegenüber treu, da sie aus eben dieser entstanden ist. Allerdings sollte

erwähnt werden, dass die Fälle $\mathcal{S} = \emptyset$ und $\mathcal{S} = \mathcal{M}$ nicht in der Stichprobenziehung enthalten sind. Stattdessen werden $Z_\mathcal{D}$ und $\nu_\mathcal{D}$ um die entsprechenden Zeilen erweitert und der Diagonalmatrix $W_\mathcal{D}$ werden ebenfalls zwei Diagonaleinträge $c$ ergänzt. Diese Prozedur führt uns zu der Approximation

$$\phi = \underbrace{(Z_\mathcal{D}^T W_\mathcal{D} Z_\mathcal{D})^{-1} Z_\mathcal{D}^T W_\mathcal{D}}_{=:R_\mathcal{D}} \cdot \nu_\mathcal{D}. \tag{4.1.5}$$

Wenn nun mehrere Vorhersagen erklärt werden sollen, besteht der Vorteil, dass die Matrix $R_\mathcal{D}$ nur einmal berechnet werden muss, falls man sich die Stichprobe $\mathcal{D}$ merkt. Tatsächlich ist dies in der Kernel-SHAP-Methode nicht der Fall: Für jede zu erklärende Instanz eines Datensatzes wird $\mathcal{D}$ erneut gezogen. Die Matrix $R_\mathcal{D}$ ist von der Form $(M+1) \times |\mathcal{D}|$. In der ergänzenden Annahme, dass $\nu_\mathcal{D}$ bereits im Vorfeld berechnet wurde, hat man nun ein leichtes Spiel: Um verschiedene Vorhersagen eines Modells zu erklären, wird nur noch eine Matrixmultiplikation benötigt:

$$R_\mathcal{D} \nu_\mathcal{D}.$$

### 4.1.2. Einfache Schätzung von $f_x$

Im zweiten Baustein der Kernel-SHAP-Methode wollen wir die Beitragsfunktion $f_x$ schätzen [AJL21]. Wenn wir den Vektor $\nu$, der die Funktion $f_x$ darstellt, exakt berechnen möchten, benötigen wir, wie im obigen Beispiel 2.1.2 auch, für jede mögliche Teilmenge $\mathcal{S}$ der Merkmale den Beitragswert $\nu(\mathcal{S})$. Wir wissen bereits, dass alle möglichen Teilmengen in der Matrix $Z$ aufgeführt sind – bei Verwendung der Stichproben $\mathcal{D}$ betrachtet man natürlich die Matrix $Z_\mathcal{D}$. Zu Beginn des dritten Kapitels wurde bereits erwähnt, dass bei Kernel SHAP Werten gilt

$$\nu(\mathcal{S}) = \mathbb{E}[f(x)|x_\mathcal{S} = x_\mathcal{S}^\star]. \tag{4.1.6}$$

Die Beitragsfunktion wird wie folgt berechnet

$$\begin{aligned} \nu(\mathcal{S}) &= \mathbb{E}[f(x)|x_\mathcal{S} = x_\mathcal{S}^\star] && (4.1.7)\\ &= \mathbb{E}[f(x_{\bar{\mathcal{S}}}, x_\mathcal{S})|x_\mathcal{S} = x_\mathcal{S}^\star] && (4.1.8)\\ &= \int f(x_{\bar{\mathcal{S}}}, x_\mathcal{S})\ p(x_{\bar{\mathcal{S}}}|x_\mathcal{S} = x_\mathcal{S}^\star)\ dx_{\bar{\mathcal{S}}}. && (4.1.9) \end{aligned}$$

Der Term $p(x_{\bar{\mathcal{S}}}|x_\mathcal{S} = x_\mathcal{S}^\star)$ gibt die bedingte Verteilung von $x_{\bar{\mathcal{S}}}$ an unter der Bedingung, dass $x_\mathcal{S} = x_\mathcal{S}^\star$. Wir können demnach leicht erkennen, dass wir zur exakten Berechnung von $\nu(\mathcal{S})$ eben diese bedingte Dichte benötigen. Diese ist allerdings selten bekannt.

An dieser Stelle in der Berechnung wird in der Kernel-SHAP-Methode eine Annahme verwendet, die womöglich weitreichende Auswirkungen mit sich bringt. Man ersetzt

die bedingte Dichte durch die unbedingte Dichte:

$$p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^{\star}_{\mathcal{S}}) \approx p(x_{\bar{\mathcal{S}}}). \tag{4.1.10}$$

Grundlage dieser Annäherung ist, dass man hier von einer Unabhängigkeit der Merkmale ausgeht. Es wird angenommen, dass zwischen den inkludierten und exkludierten Merkmalen, genauer zwischen $x_{\mathcal{S}}$ und $x_{\bar{\mathcal{S}}}$, keine Abhängigkeit besteht. Tatsächlich weisen viele Datensätze durchaus Abhängigkeiten in ihren Strukturen auf. Diese aufzubrechen bzw. zu ignorieren könnte durchaus fehlerhafte Ergebnisse zur Folge haben.

Das in Gleichung 4.1.9 stehende Integral kann nun durch Monte Carlo Integration approximiert werden:

$$\nu_{\text{KernelSHAP}}(\mathcal{S}) = \frac{1}{K}\sum_{k=1}^{K} f(x^{k}_{\bar{\mathcal{S}}}, x^{\star}_{\mathcal{S}}). \tag{4.1.11}$$

Dabei wird der Datensatz, der zum Trainieren des Modells $f$ verwendet wurde, als empirische Verteilung von $x$ interpretiert. Die Vektoren $x^{k}_{\bar{\mathcal{S}}}$ mit $k = 1, ...K$ stehen für Stichproben aus dem Hintergrunddatensatz bei Betrachtung der Teilmenge $\bar{\mathcal{S}}$ und damit ohne Berücksichtigung der Merkmale $x_{\mathcal{S}}$. Diese wurden also aufgrund der angenommenen Unabhängigkeit frei von Abhängigkeiten gezogen.

Mit dieser Schätzung kann jetzt in der Approximation der Shapley-Werte in den ersten Baustein eingesetzt werden.

## 4.2. Implementierter Algorithmus – Vorgehen der Stichprobenziehung

Für Python existiert bereits ein Paket ‚SHAP' [Lun22], das den wichtigen SHAP-Methoden einen anwendbaren Rahmen gibt [Lun18]. Dieses wurde anlässlich des Papers von Lundberg und Lee [LL17] veröffentlicht und vereinigt verschiedene ‚Explainer' (zu Deutsch, Erklärer), anschauliche Aufbereitungsmethoden und Diagramme sowie einiges mehr [Lun18]. Die Parallelen zwischen den zwei besprochenen Bausteinen und dem Code sind nicht ganz offensichtlich, was das Nachvollziehen des nächsten Kapitels erschwert. Aus diesem Grund diskutieren und veranschaulichen wir in diesem Abschnitt den in dem Paket implementierten Schritt der Stichprobenziehung.

Nach dem eigentlichen Shapley-Verfahren berechnen wir den Ertrag einer jeden Koalition mit und ohne einen jeden Spieler. Wir möchten also, wie zuvor auch, verschiedene Teilmengen von Merkmalen durch das Modell laufen lassen, um den Output zu ermitteln.

Das bringt zwei Probleme mit sich:

1. Das Ablesen des Ertrags bei ‚Weglassen' eines Merkmals ist häufig bei Modellen nicht ohne Weiteres möglich, da unklar ist, wie man ein Merkmal ausblenden kann.

2. Wie bereits erwähnt sorgt eine große Anzahl an Merkmalen für viele aufwendige Rechenoperationen.

Zunächst benötigen wir also einen Weg, der das Aussetzen eines Features vortäuscht, damit die Ergebnisse des Modells ‚ohne' dieses Merkmal beobachtet werden können. Wir veranschaulichen das Verfahren an einem Beispiel:

**Beispiel 4.2.1.** *An einem warmen Sommertag möchten wir den Einfluss verschiedener Faktoren auf die Anzahl der verkauften Eis-Kugeln in einem Schwimmbad ermitteln.*
*Als Input-Parameter beobachten wir die Anzahl der verschiedenen Eissorten (E), der zu mietenden Liegen (L), die Anzahl der Sonnenstunden (S) und die Temperatur (T). Wir haben bereits ein trainiertes Modell und wollen nun für den morgigen Tag* $x = (12, 60, 14, 28)$ *bestimmen, welchen Einfluss die* 12 *verschiedenen Eissorten,* 60 *zu mietenden Liegen,* 14 *Sonnenstunden und* $28°C$ *auf die Verkaufszahlen haben.*

*Das Vorgehen im Code lässt sich wie folgt darstellen: Wir möchten gerne verschiedene Teilmengen der Merkmale an das Modell übergeben. Nur verhindert das Modell das Ausblenden eines Merkmals. Um das Problem zu umgehen, generieren wir einen Hintergrunddatensatz, der im Normalfall aus den Trainingsdaten besteht. Wenn wir nun die Koalition aus Eissorten, Sonnenstunden und Temperatur dem Modell übergeben, wir also das zweite Merkmal, die Liegen, ausblenden möchten, setzen wir in dem Hintergrunddatensatz die gesamte erste Spalte auf* 12*, die zweite Spalte bleibt unberührt – entspricht demnach weiterhin den Trainingsdaten, die dritte Spalte setzen wir auf* 14 *und die vierte auf* 28*.*

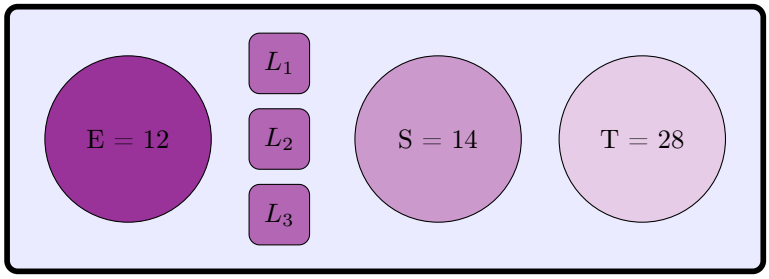
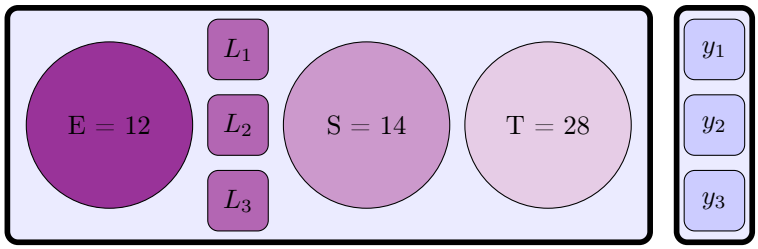



Abbildung 4.1.: **Modifikation des repräsentativen Datensatzes in der originalen Kernel-SHAP-Methode.** Die drei Kreise $E$, $S$ und $T$ stehen für die festgesetzten Daten, also diejenigen, die in der Teilmenge enthalten sind. Die drei Vierecke sind Platzhalter für die Werte der Liegen im repräsentativen Datensatz. Natürlich stehen letztere stellvertretend für eine Vielzahl an Daten.

*Wir könnten die Situation also so beschreiben, dass wir alle Merkmale – außer jene, die wir auslassen möchten – festsetzen. Diesen Datensatz können wir nun dem Modell übergeben und bekommen für jede Zeile in dem modifizierten Hintergrunddatensatz ein Ergebnis. Um einen Konsens in diesen Outputs zu schaffen, bilden wir über die einzelnen Vorhersagen des Modells den Erwartungswert. Dieser wird mit der Formel $\nu_{KernelSHAP}$ in 4.1.11 berechnet.*




$$y_{1i34} = \mathbb{E}(y_j \text{ für alle } j)$$

Abbildung 4.2.: **Ergebnisauswertung des modifizierten Hintergrunddatensatzes in der originalen Kernel-SHAP-Methode.** Wir betrachten eine ähnliches Bild wie in Abbildung 4.1. An der rechten Seite sehen wir angefügt die Modell-Outputs der jeweiligen Zeile im modifizierten Datensatz. Untenstehend den Erwartungswert über diese. Diesen Wert nennen wir $y_{1i34}$, um zu signalisieren, dass das zweite Merkmal nicht festgesetzt ist.

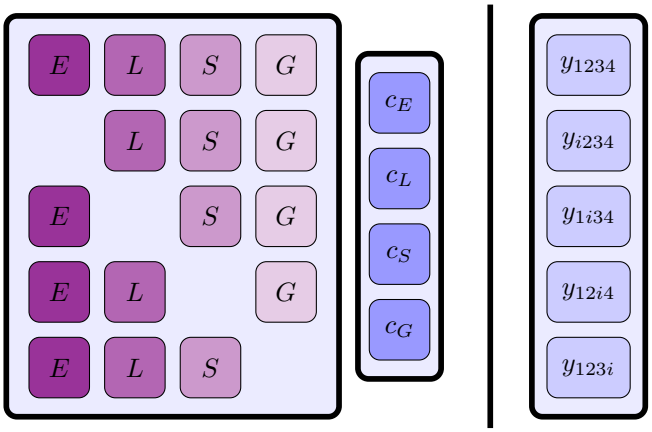



Abbildung 4.3.: **Gleichungssystem des modifizierten Hintergrunddatensatzes in der originalen Kernel-SHAP-Methode.** Aus der vorherigen Abbildung bauen wir dieses Gleichungssystem auf. Linksseitig ist zunächst die Auflistung der betrachteten Teilmengen und dann einen Vektor, der durch das Gleichungssystem berechnet werden soll und jedem Merkmal den entsprechenden Beitrag zuordnen soll, zu finden. Auf der rechten Seite sind die zuvor bestimmten Erwartungswerte der einzelnen Teilmengen platziert.

*Das Wissen über die erwarteten Outputs der einzelnen Teilmengen wird verwendet, um das Gleichungssystem in Abbildung 4.3 aufzustellen. Bei genauer Betrachtung dieses wird deutlich, dass die beschriebene Abbildung 4.2 hier in der dritten Zeile wiederzufinden ist.*

*Besonderer Erwähnung bedarf, dass nicht alle möglichen Koalitionen in dem Gleichungssystem auftauchen. Vielmehr ist dies hier die Stelle, an der wir die Stichproben $\mathcal{D} \subset \mathcal{M}$ ziehen, was in Unterkapitel 4.1.1 genauer beschrieben ist. Ebenfalls hier können wir durch die Ziehung von Stichproben Rechenkomplexität einsparen.*

*Wenn wir dieses Gleichungssystem jetzt den Shapley-Kernel-Gewichten*

$$\frac{\#\ Merkmale\ -\ 1}{\#\left\{\begin{smallmatrix} Koalitionen \\ der\ Größe\ |C| \end{smallmatrix}\right\} \cdot \#\left\{\begin{smallmatrix} eingeschlossene \\ Merkmale\ in\ C \end{smallmatrix}\right\} \cdot \#\left\{\begin{smallmatrix} ausgeschlossene \\ Merkmale\ aus\ C \end{smallmatrix}\right\}} \tag{4.2.1}$$

*entsprechend lösen, gibt der Vektor c die SHAP-Werte an. Damit ordnet er jedem Merkmal einen Beitrag zu.*

Noch einmal zusammen gefasst durchlaufen wir also die nachstehenden Schritte:

1. Wir generieren einen Hintergrunddatensatz, der im Normalfall aus den Trainingsdaten besteht. Die Wahl dieses repräsentativen Datensatzes spielt ebenfalls eine große Rolle bei den folgenden Modifikationen.

2. Um das Weglassen der Merkmale zu simulieren, die nicht in der Teilmenge enthalten sein sollen, wenden wir einen ‚Trick' an: Wir setzen in dem Hintergrunddatensatz alle Merkmale, die in der Teilmenge enthalten sind, auf ihren Wert im zu erklärenden Datensatz fest.

3. Für diesen Datensatz bestimmen wir die jeweiligen Modellvorhersagen und die Erwartungswerte über all diese je Teilmenge.

4. Wir stellen ein Gleichungssystem auf, in dem auf der linken Seite die einzelnen Teilmengen mit einem unbekannten Vektor $c$, der die Beiträge der einzelnen Merkmale angeben soll, multipliziert wird und auf der rechten Seite die zuvor bestimmten Erwartungswerte je Teilmenge aufgeführt sind.

5. Dieses wird den Gewichten in Formel 4.2.1 entsprechend gelöst.
   Der Vektor $c$ gibt die SHAP-Werte an.

# 5. Modifiziertes Kernel SHAP

Die Modifizierungen in diesem Kapitel und die dazugehörigen Experimente im nächsten Kapitel stellen den wesentlichen Kern der Bachelorarbeit dar. Zunächst betrachten wir den formellen Rahmen, in den das neue Skript eingepflegt wurde und sprechen über wichtige und interessante Punkte bei der Implementierung. Danach werden die drei bzw. vier einzelnen Methoden beschrieben – je einmal aus der mathematischen Perspektive und einmal dem Vorgehen im Code entsprechend. Insbesondere bei der mathematischen Betrachtung der Modifizierungen orientieren wir uns maßgeblich an Aas et al. [AJL21].

## 5.1. Allgemeines

Wie bereits im vorherigen Kapitel erwähnt, bildet die Annahme der Unabhängigkeit zwischen den Merkmalen die Grundlage der Approximation der bedingten Dichte. Dass Unabhängigkeit zwischen sämtlichen Merkmalen in Datensätzen besteht, ist dabei eher die Ausnahme. Typische Beispiele wären Quadratmeterzahl und Zimmeranzahl bei der Wohnungspreisschätzung, Temperatur und Anzahl der Sonnenstunden bei der Verkaufszahlenschätzung von Eiscreme oder Übergewicht und ungesunde Ernährungsgewohnheiten bei der Risikoeinschätzung für Brustkrebs. Es erscheint demnach sinnvoll, in die Kernel-SHAP-Methode Abhängigkeiten miteinzubeziehen, um realistischere Ergebnisse zu liefern.

Obwohl die Kernel-SHAP-Methode durchaus die Möglichkeit bietet, hochkomplexe Datenstrukturen wie Bilder und Texte zu erklären, beschränken wir uns in den Modifizierungen auf die Anwendung bei tabellarischen Daten.

### Die richtige Stelle

Bei genauerer Betrachtung des in Abschnitt 4.1.2 vorgestellten Vorgangs wird klar, dass

$$p(x_{\bar{S}}|x_S = x_S^\star) \approx p(x_{\bar{S}})$$

die einzige Stelle ist, an der die Unabhängigkeitsannahme gebraucht wird. Statt dieser Approximation, werden im Folgenden verschiedene Ansätze vorgestellt, welche die bedingte Dichte $p(x_{\bar{S}}|x_S = x_S^\star)$ korrekt schätzen.

### Implementierung

Das Python-Paket SHAP enthält die Kernel-SHAP-Methode vollständig implementiert und bietet uns damit die Grundlage die nachstehenden Modifizierungen einzu-

bauen. Da es das Vorgehen aus Abschnitt 4.1 nachbildet, nimmt es wie in Gleichung 4.1.10 beschrieben auch die Unabhängigkeit der einzelnen Merkmale an. Diese Annahme möchten wir gerne fallen lassen und stellen dazu die in dem dieser Arbeit zugrundeliegenden Hauptpaper [AJL21] angeführten Ansätze vor.

Inhaltlich bedeutet das für uns, dass das existierende SHAP-Paket um eine Klasse, die der ‚Explainer'-Klasse ‚Kernel' untersteht, von uns erweitert wurde. Sie enthält fünf Unterklassen für neue Ansätze, die zum Teil voneinander und von der originalen Klasse erben. Als eine besondere Schwierigkeit stellte sich dabei heraus, die richtige Struktur mit den passenden Stellen in zuvor fremden Code abzupassen, um Redundanzen zu vermeiden, die das Programm zusätzlich bewältigen müsste.

Im Anhang ist in Abbildung A.1 zwecks besserer Übersicht ein Ablaufdiagramm der vereinfachten Schritte des ‚KernelExplainer' zur Berechnung der SHAP-Werte im Python-Paket dargestellt.

## 5.2. Multivariaten Gaußverteilung

Für die erste Modifikationsmöglichkeit wird angenommen, dass der Vektor $x$ einer multivariaten Gaußverteilung mit Erwartungswert $\mu$ und Kovarianz $\Sigma$ entspricht. Ebenfalls ist $p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^{\star}_{\mathcal{S}})$ eine Gaußverteilung.

### Mathematische Betrachtung

Wir stellen

$$p(x) = p(x_{\mathcal{S}}, x_{\bar{\mathcal{S}}}) = \mathcal{N}_M(\mu, \Sigma) \tag{5.2.1}$$

dar mit den Parametern $\mu$ und $\Sigma$ der Form

$$\mu = \begin{pmatrix} \mu_{\mathcal{S}} \\ \mu_{\bar{\mathcal{S}}} \end{pmatrix} \text{ und } \Sigma = \begin{pmatrix} \Sigma_{\mathcal{S}\mathcal{S}} & \Sigma_{\mathcal{S}\bar{\mathcal{S}}} \\ \Sigma_{\bar{\mathcal{S}}\mathcal{S}} & \Sigma_{\bar{\mathcal{S}}\bar{\mathcal{S}}} \end{pmatrix}. \tag{5.2.2}$$

Daraus kann man schließen, dass

$$p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^{\star}_{\mathcal{S}}) = \mathcal{N}_{|\bar{\mathcal{S}}|}(\mu_{\bar{\mathcal{S}}|\mathcal{S}}, \Sigma_{\bar{\mathcal{S}}|\mathcal{S}}). \tag{5.2.3}$$

Diese Verteilung folgt den Parametern $\mu_{\bar{\mathcal{S}}|\mathcal{S}}$ und $\Sigma_{\bar{\mathcal{S}}|\mathcal{S}}$, die berechnet werden durch:

$$\mu_{\bar{\mathcal{S}}|\mathcal{S}} = \mu_{\bar{\mathcal{S}}} + \Sigma_{\bar{\mathcal{S}}\mathcal{S}}{\Sigma_{\mathcal{S}\mathcal{S}}}^{-1}(x^{\star}_{\mathcal{S}} - \mu_{\mathcal{S}}) \text{ und} \tag{5.2.4}$$

$$\Sigma_{\bar{\mathcal{S}}|\mathcal{S}} = \Sigma_{\bar{\mathcal{S}}\bar{\mathcal{S}}} - \Sigma_{\bar{\mathcal{S}}\mathcal{S}}\Sigma^{-1}_{\mathcal{S}\mathcal{S}}\Sigma_{\mathcal{S}\bar{\mathcal{S}}}. \tag{5.2.5}$$

Diese Ergebnisse entsprechen den üblichen Eigenschaften der mehrdimensionalen Normalverteilung bzw. der bedingten Verteilung bei partieller Kenntnis des Zufallsvektors $x = (x_{\bar{\mathcal{S}}}, x_{\mathcal{S}})$. Bei Interesse an der Herleitung dieser Parameter und weiteren Informationen wird auf das Werk von Rasmussen und Williams [RW06] verwiesen.

Wenn $\mu$ und $\Sigma$ durch die Trainingsdaten gegeben sind, was bedeutet, dass mithilfe dieser Daten der Mittelwert und die Kovarianz bestimmt wird, dann kann eine Stichprobe von der Gaußverteilung mit Erwartungswert $\mu_{\bar{\mathcal{S}}|\mathcal{S}}$ und Kovarianz $\Sigma_{\bar{\mathcal{S}}|\mathcal{S}}$ gezogen werden statt von der Verteilung $p(x_{\bar{\mathcal{S}}})$.
Wieder gilt, dass durch die Stichproben der bedingten Verteilung $x^k_{\bar{\mathcal{S}}}$ mit $k = 1, ...K$ das Integral in (4.1.9) durch den Term in (4.1.10) approximiert wird.

### Vorgehen im Code

Im Code wird dieser Ansatz implementiert, indem wir den Schritt des Strichprobenziehens modifizieren. Im Vergleich zu dem beschriebenen Vorgehen der Originalmethode wird, kurz formuliert, ein anderer Hintergrunddatensatz gewählt. Statt die Trainingsdaten zu verwenden, um die Vorhersagen beim Auslassen verschiedener Merkmale zu ermitteln, verwenden wir in dieser Variante zufällig erzeugte Stichproben, die einer multivariaten Gaußverteilung folgen. Wir bestimmen zu Beginn den Erwartungswert und die Kovarianz des Trainingsdatensatzes. Den bedingten Erwartungswert und die bedingte Kovarianz berechnen wir nach den oben stehenden Formeln 5.2.4 und 5.2.5. Falls $\Sigma_{\mathcal{SS}}$ nicht invertierbar ist, verwenden wir an dieser Stelle die Pseudo-Inverse.

Diese neuen Werte werden verwendet, um Stichproben einer multivariaten Gaußverteilung mit dem neuen, bedingten Erwartungswert und der neuen, bedingten Kovarianz als Parameter zu ziehen, die wiederum als Hintergrunddatensatz verwendet werden. Von da an wird wie im originalen Kernel SHAP verfahren. Dieser Hintergrunddatensatz wird also an das Modell übergeben, über dessen Ergebnisse der Erwartungswert gebildet wird und mit diesem das Gleichungssystem aus Abbildung 4.3 aufgestellt und gelöst wird. Wieder gibt der Vektor $c$ die Shapley-Werte der einzelnen Merkmale an.

## 5.3. Gauß-Copula

Wenn der Vektor $x$, der die einzelnen Merkmale beinhaltet, keine multivariate Gaußverteilung vorweist, kann man die Randverteilungen durch ihre empirische Verteilung modellieren. Auf diesem Wege kann die Abhängigkeit mithilfe der Gauß-Copula dargestellt werden.

### Mathematische Betrachtung

Copulas dienen der Modellierung von Abhängigkeitsstrukturen multivariater Verteilungsfunktionen basierend auf ihren Randverteilungen. Eine gemeinsame Wahrscheinlichkeitsverteilung kann zum einen in die Randverteilungen und zum anderen in eine Funktion zerlegt werden, die diese miteinander verbindet. Diese ‚Kopplungsfunktion ‘ ist die Copula und ermöglicht es, die Korrelation separat anzugeben.
In vielen verschiedenen Bereichen, besonders in der Finanz- und Versicherungsmathematik, finden sie Anwendungen. Da dieses Thema weit über die Grenzen der vor-

liegenden Arbeit ausgearbeitet werden könnte, wird es im Folgenden nur begrenzt angeschnitten.

Mehr zu diesem Thema ist beispielsweise in den allgemeinen Werken von Nelson [Nel07], Freez und Valdez [FV98] oder anwendungsbezogen im Bereich der Modellierung des Risikomanagements von Embrechts et al. [ELM01] zu finden.

Vereinfacht formuliert wird die Gauß-Copula genutzt, um die gemeinsame Korrelation verschiedener Randverteilungen, die selber durch Normalverteilungen beschrieben werden können, zu berechnen, wenn diese nicht unabhängig sind.

**Definition 5.3.1** (**Copula**)**.** Eine multivariate Verteilungsfunktion
$C : [0,1]^n \to [0,1]$ nennt man Copula, wenn ihre einzelnen Randverteilungen gleichverteilt auf dem Intervall $[0,1]$ sind.

Das folgende Theorem ist nützlich, um die Intuition dahinter zu verstehen [Skl59].

**Satz 5.3.2** (Sklars Theorem)**.** *Eine multivariate Verteilungsfunktion*
$F : \{\mathbb{R} \cup \pm\infty\}^n \to [0,1]$ *mit Randverteilungen* $F_1, ..., F_n : \mathbb{R} \cup \{\pm\infty\} \to [0,1]$ *kann mithilfe einer n-dimensionalen Copula C dargestellt werden durch*

$$F(x_1, ..., x_n) = C(F_1(x_1), ...F_n(x_n)) \tag{5.3.1}$$

*für alle* $(x_1, ...x_n) \in \mathbb{R} \cup \{\pm\infty\}$.

Die Gleichung (5.3.1) kann geschrieben werden als

$$C(u_1, ..., u_n) = F(F_1^{-1}(u_1), ..., F_n^{-1}(u_n)), \tag{5.3.2}$$

wobei $F_j^{-1}$ den inversen Funktionen der Randverteilungen entspricht.

Um nun Stichproben von $p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x_{\mathcal{S}}^\star)$ zu ziehen, folgen wir dem nachstehenden Prozedere:

1. Zunächst überführen wir die einzelnen Randverteilungen $X_j$ von der Verteilung der Merkmale $X$ zu einer gauß'schen Merkmalsverteilung $V_j$, indem wir die inverse Gaußverteilung auf die empirische Verteilungsfunktion $\hat{F}_j$ der $j$-ten Randverteilung anwenden:

$$V_j = \Phi^{-1}(\hat{F}_j(X_j)).$$

2. Nun nehmen wir an, dass $V$ multivariat normalverteilt ist. Demnach können wir unsere Methode aus dem ersten Ansatz verwenden: Wir bestimmen zu Beginn den Erwartungswert $\mu$ und die Kovarianz $\Sigma$, daraufhin die bedingten Parameter und können von der bedingten Verteilung $p(v_{\bar{\mathcal{S}}}|v_{\mathcal{S}} = v_{\mathcal{S}}^\star)$ basierend auf diesen Parametern Stichproben nehmen.

3. Im letzten Schritt konvertieren wir die gezogenen Stichproben der konstruierten Randverteilungen $V_j$ wieder zu original verteilten Werten zurück:

$$\hat{X}_j = \hat{F}_j^{-1}(\Phi(V_j)).$$

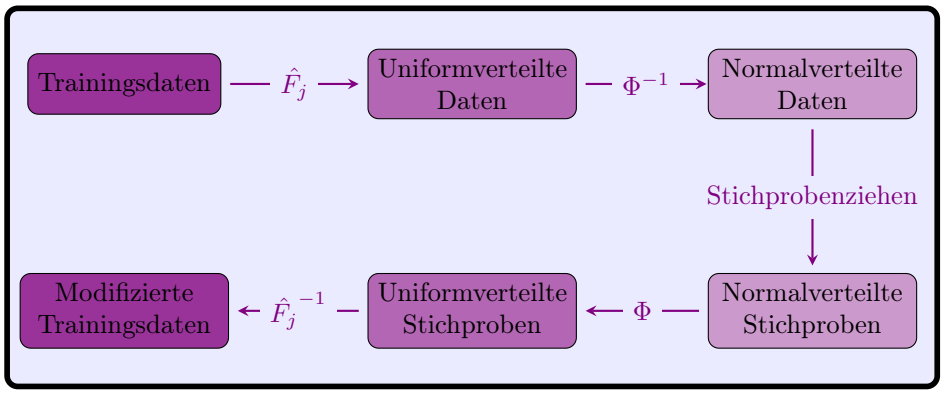



Abbildung 5.1.: **Vorgang des Stichprobenziehens beim Copula-Ansatz.** Die zuvor stehenden drei Schritten zur Anwendung der Copula bildlich veranschaulicht.

Wie in der vorherigen Methode gilt, dass auf diesem Wege das Integral in (4.1.9) durch den Term in (4.1.10) approximiert wird.

### Vorgehen im Code

Wieder setzen wir an der Stelle im Code ein, in der der Hintergrunddatensatz erzeugt wird. In diesem Fall wenden wir auf den Trainingsdatensatz die empirische Randverteilung an. Für jedes Merkmal, also jede Spalte einzeln, werden die entsprechenden neuen Werte zwischen Null und Eins ermittelt. Es ergibt sich damit ein uniformverteilter Datensatz. Im nächsten Schritt wendet man die inverse Standardnormalverteilung darauf an. Es ergibt sich ein Datensatz, der normalverteilt ist.

Hier können wir wieder in der vorherigen Methode einsetzen: Wir bestimmen den bedingten Erwartungswert und die bedingte Kovarianz und ziehen auf Basis dieser Parameter Stichproben. Diese neuen Stichproben müssen nun allerdings zurücktransformiert werden, der vorherige Prozess wird rückgängig gemacht. Das bedeutet, wir wenden die Standardnormalverteilung auf diesen Datensatz an, um einen uniformen Datensatz zu erzeugen. Daraufhin wird die inverse, spaltenweise Randverteilung auf den Datensatz angewandt. Wir erhalten schlussendlich also die Stichproben, die der ursprünglichen Verteilung folgen. Diese werden als Hintergrunddatensatz verwendet. Von nun an setzen wir wieder in der originalen Kernel-SHAP-Methode ein. Wie auch beim ersten Ansatz verwendet man den neuen Hintergrunddatensatz, um den Erwartungswert über die Vorhersagen des Modells zu erzeugen und dann das Gleichungssystem in Abbildung 4.3 aufzubauen. Auch hier entspricht der Lösungsvektor $c$ den gesuchten Shapley-Werten.

## 5.4. Empirische bedingte Verteilung

Wenn nun die beiden ersten Verfahren nicht greifen können, da weder eine Normalverteilung der Ränder gegeben ist, noch eine gauß'sche Abhängigkeitsstruktur im Allgemeinen, können wir uns mit einem anderen Ansatz behelfen: Wir bedienen uns den Möglichkeiten einer nicht-parametrischen Schätzung.
Ein klassisches, nicht-parametrisches Verfahren zur Schätzung der Wahrscheinlichkeitsverteilung einer Zufallsvariable ist die sogenannte ‚Kerndichtenschätzung'. Diese wird begleitet von einem Nachteil, den viele nicht-parametrische Schätzungen mit sich bringen: dem Fluch der Dimensionalität. Je höher die Dimensionen, also je mehr Merkmale verwendet werden, desto komplexer wird sie in der Anwendung. Eine weitere Anforderung an das Verfahren besteht darin, dass Stichproben aus der geschätzten Verteilung erzeugt werden sollen.

Im Folgenden wird ein empirischer Ansatz vorgestellt, um eine Stichprobe aus $p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^\star_{\mathcal{S}})$ zu ziehen.

Neben der hier vorgestellten Methode, bei der $\sigma$ und $\mu$ festgesetzt sind oder dem Programm übergeben werden können, stellen Aas et al. eine weitere Version vor: Basierend auf dem ‚Akaike information criterion', das von Hurvich et al. entwickelt wurde [HST98], können die benötigten Parameter bestimmt werden. Da diese zusätzliche Version der dritten Methode aufwendig ist, haben wir sie für diese Arbeit außen vor gelassen. Für weitere Informationen wird auf Aas et al. verwiesen [AJL21].

### Mathematische Betrachtung

Basierend auf der Idee, dass Stichproben $(x_{\bar{\mathcal{S}}}, x_{\mathcal{S}})$, welche die Eigenschaft aufweisen, dass $x_{\mathcal{S}}$ nah an $x^\star_{\mathcal{S}}$ liegt, informativer bezüglich der bedingten Dichte $p(x_{\bar{\mathcal{S}}}|x_{\mathcal{S}} = x^\star_{\mathcal{S}})$ sind, durchläuft man die nachstehenden Schritte:

1. Für jede Zeile in dem Datensatz $x^i$ wird die Distanz zu der zu erklärenden Instanz $x^\star$ bestimmt. Wir verwenden dabei eine abgewandelte Mahalanobis-Distanz [Mah36], die sich in diesem Zusammenhang erprobt hat:

$$D_{\mathcal{S}}(x^\star, x^i) = \sqrt{\frac{(x^\star_{\mathcal{S}} - x^i_{\mathcal{S}})^T \Sigma_S^{-1} (x^\star_{\mathcal{S}} - x^i_{\mathcal{S}})}{|\mathcal{S}|}}. \tag{5.4.1}$$

   Dabei steht $\Sigma_{\mathcal{S}}^{-1}$ für die inverse Kovarianz innerhalb des Trainingsdatensatzes von $x_{\mathcal{S}}$. Wir betrachten also bei der Berechnung der Distanz nur die Merkmale, die in der betrachteten Teilmenge $|\mathcal{S}|$ enthalten sind.

2. Diese Distanz wird nun verwendet, um jeder Zeile des Trainingsdatensatzes ein Gewicht ähnlich zur Gaußverteilung zuzuordnen:

$$w_{\mathcal{S}}(x^\star, x^i) = \exp\left(-\frac{D_{\mathcal{S}}(x^\star, x^i)^2}{2\sigma^2}\right). \tag{5.4.2}$$

Das $\sigma$ ist ein Parameter, welcher der Glättung dient. Dieser muss näher spezifiziert oder nach dem Vorbild von [AJL21] und [HST98] bestimmt werden.

3. Innerhalb des Trainingsdatensatzes wählen wir nun die $K$ am höchsten gewichteten Zeilen. Die Anzahl der verwendeten Instanzen $K$ wird dabei durch den folgenden Term bestimmt:

$$K = \min_{L \in \mathbb{N}} \left\{ \frac{\sum_{k=1}^{L} w_{\mathcal{S}}(x^\star, x^k)}{\sum_{i=1}^{n} w_{\mathcal{S}}(x^\star, x^i)} > \eta \right\}. \tag{5.4.3}$$

   Das $\eta$ wird auf z. B. 0.9 festgesetzt. Wenn das gerade berechnete $K$ größer als ein gewisser Wert wird, setzen wir es ebenfalls auf einen Wert, beispielsweise 5000 fest.

4. Nun berechnet man ähnlich wie in 4.1.11 die Shapley-Werte mit der folgenden Formel:

$$\nu_{\text{EmpCond}}(\mathcal{S}) = \frac{\sum_{k=1}^{K} w_{\mathcal{S}}(x^\star, x^k) f(x_{\bar{\mathcal{S}}}, x^\star_{\mathcal{S}})}{\sum_{k=1}^{K} w_{\mathcal{S}}(x^\star, x^k)}. \tag{5.4.4}$$

   Wir verwenden also zur Berechnung der Shapley-Werte nur noch die $K$ wichtigsten Instanzen.

### Vorgehen im Code

In der Implementierung dieses Ansatzes gehen wir folgendermaßen vor: Wieder möchte man einen Hintergrunddatensatz bestimmen. Für jede Zeile im Trainingsdatensatz wird die Distanz und das darauf aufbauende Gewicht bestimmt. Die Instanzen werden den Gewichten entsprechend absteigend sortiert und das $K$ berechnet, indem die Gewichte addiert werden bis sie $\eta$ erreicht haben. Nur die Instanzen, die zu den $K$ wichtigsten gehören, werden nun dem Hintergrunddatensatz zugeordnet, alle anderen werden auf null gesetzt.

Anders als bei den vorherigen Ansätzen muss allerdings ebenfalls die Berechnung des Erwartungswertes angepasst werden. Die oben in 5.4.2 berechneten Gewichte werden in Abhängigkeit von dem in 5.4.3 berechnetem $K$ normiert und der Erwartungswert um diese normierten Gewichte ergänzt.

## 5.5. Kombinierung zweier Ansätze

Die Experimente in dem Hauptpaper von Aas et al. [AJL21] ergaben, dass verschiedene Methoden für verschiedene Dimensionen von $x_{\mathcal{S}}$, also verschiedene Teilmengengrößen unterschiedlich gut abschneiden. Ähnliches wurde bereits in vorangehenden Werken beobachtet, wie z. B. von Izbicki und A. B. Lee [IL17] und darin enthaltenen Verweisen. Während der dritte Ansatz, die empirisch bedingte Verteilung, bei kleineren Dimension $dim(x_{\mathcal{S}}) \leq D^\star$ bessere Ergebnisse liefert, schneiden die ersten

beiden Methoden, die Multivariaten Gaußverteilung und die Gauß-Copula, bei höheren Dimensionen besser ab.

### Mathematische Betrachtung

Aus dieser Erkenntnis entwickeln sich zwei neue Methoden:

1. Eine Kombination aus der Multivariaten Gaußverteilung und der empirisch bedingten Verteilung (Erster und dritter Ansatz)

2. Die Kombination aus der Gauß-Copula und der empirisch bedingten Verteilung (Zweiter und dritter Ansatz)

Obwohl sie kaum als eigene Methoden gelten, werden wir sie nachfolgend zur besseren Unterscheidung häufig mit ‚vierter' und ‚fünfter' Methode betiteln.

### Vorgehen im Code

Bei der Anschauung des Codes bedeutet dies, dass für jede Teilmenge aufs Neue entschieden wird, ob die erste bzw. zweite Methode oder die dritte Methode verwendet wird. Wenn die Größe der betrachteten Teilmenge kleiner gleich $D^\star$ ist, verwenden wir den Ansatz der empirisch bedingten Verteilung und berechnen auch dementsprechend den Erwartungswert $\nu_{\text{EmpCond}}$. Andernfalls wählen wir den ersten beziehungsweise zweiten Ansatz, je nach Kombination.

# 6. Experimente

Zu untersuchen, ob die Modifizierungen besser sind, also zutreffendere Ergebnisse liefern, als die originale Methode, ist kompliziert. Eigentlich ist eine Validierung dieser Art nur dann möglich, wenn man die ‚wahren' Shapley-Werte kennt, um die Ergebnisse der Kernel-SHAP-Methode und unserer Modifikationen mit den Shapley-Werten zu vergleichen.

Allerdings sollte an dieser Stelle betont werden, dass es die ‚wahren' Shapley-Werte nur dann gibt, wenn man sich zuvor auf eine Funktion $f_x$ geeinigt hat. Die Auffassung, welche Funktion verwendet werden soll, um den einzelnen Teilmengen eine Vorhersage des Modells zuzuordnen, damit ein sinnvolles Ergebnis entsteht, variiert allerdings je nach Betrachter und mit Sicherheit auch nach der zu betrachtenden Situation.

Um nun trotzdem eine fundierte Validierung der neuen Methoden durchführen zu können, kreieren wir künstliche Daten einer multivariaten Gauß-Verteilung mit einmal drei und einmal sechs Merkmalen als Input-Daten $X$ und kombinieren diese mit einem Sampling-Modell und einem Vorhersagenmodell. Da wir die Daten künstlich nach einer bestimmten Verteilung generiert haben, kennen wir diese und ihre Parameter und einigen uns für die Funktion, die uns das Weglassen einiger Merkmale ermöglicht, auf den bedingten Erwartungswert. Wir betrachten demnach die Funktion $f_x = \mathbb{E}[f(x)|x_{\mathcal{S}} = x^\star_{\mathcal{S}}]$, um unsere ‚wahren' Shapley-Werte zu berechnen.

Da die exakte Berechnung der Shapley-Werte einen hohen Rechenaufwand mit sich bringt, beschränken wir uns für die Validierung mit $M = 3$ Merkmalen auf die ersten drei Methoden. Für die Validierung mit $M = 6$ wählen wir alle neu eingeführten Methoden – unter anderem die vierte und fünfte Methode. Dabei orientieren wir uns an dem Vorgehen von Aas et al. [AJL21]. Sie verwenden die Dimensionen $M = 3$ und $M = 10$ und kombinieren diese mit verschiedenen Verteilungen, diversen Sampling-Modellen und Vorhersagenmodellen. Einen Teil ihrer Experimente bilden wir nachfolgend nach.

## 6.1. Aufbau der Experimente

In unseren Experimenten ziehen wir $n_{train} = 1600$ Trainingsdaten nach einer Verteilung, mithilfe derer wir unser Modell anpassen. Dieses verwenden wir daraufhin, um unsere $n_{test} = 400$ Testdaten zu erklären.

Wir vergleichen die folgenden Methoden:

- Kernel SHAP mit der multivariaten Gaußverteilung
- Kernel SHAP mit der Gauß-Copula
- Kernel SHAP mit der empirisch bedingten Verteilung (wobei $\sigma = 0.1$ und $\eta = 0.9$ festgesetzt wird)
- Kernel SHAP mit der Kombination aus multivariater Gaußverteilung für die Teilmengen der Größe $> 3$ und empirisch bedingter Verteilung ($\sigma = 0.1$ und $\eta = 0.9$) für kleine Teilmengen
- Kernel SHAP mit der Kombination aus Gauß-Copula für die Teilmengen der Größe $> 3$ und empirisch bedingter Verteilung ($\sigma = 0.1$ und $\eta = 0.9$) für kleine Teilmengen

### 6.1.1. Vergleichsmaße

Wir führen nach Aas et al. einen Fehler ein [AJL21]:

**Definition 6.1.1** (**Mittlerer absoluter Fehler**)**.** Der mittlere absolute Fehler (Mean absolut error, MAE) zwischen den SHAP-Werten, die durch eine Methode $q$ bestimmt wurden, und den ‚wahren' Shapley-Werten ist der Form

$$\mathrm{MAE}(q) = \frac{1}{M \cdot n_{test}} \sum_{j=1}^{M} \sum_{i=1}^{n_{test}} |\Phi_{j,true}^{(i)} - \Phi_{j,q}^{(i)}|. \tag{6.1.1}$$

Dabei entspricht $M$ der Anzahl der Merkmale, $n_{test}$ der Anzahl der Daten im Testdatensatz, $\Phi_{j,true}^{(i)}$ den ‚wahren' Shapley-Werten und $\Phi_{j,q}^{(i)}$ den SHAP-Werten der Methode $q$ für das $j$-te Merkmal in der $i$-ten Vorhersage.

Diesen Fehler können wir Gneiting und Raftery [GR07] nach verwenden, um einen Skill Score aufzustellen, der eine bessere Übersicht über das Können der einzelnen Methoden bietet.

**Definition 6.1.2** (**Skill Score**)**.** Für die Methode $q$ ist der Skill Score basierend auf den zuvor bestimmten MAEs definiert durch

$$\mathrm{Skill}(\mathrm{MAE}, q) = 1 - \frac{\mathrm{MAE}(q)}{\mathrm{MAE}(original)}. \tag{6.1.2}$$

$\mathrm{MAE}(original)$ beschreibt den MAE der originalen Kernel-SHAP-Methode.

## 6.2. Experimente für drei Merkmale

Da die vierte und fünfte Methode nach der Größe der Teilmenge entscheidet und mehr als drei Merkmale bei insgesamt drei Merkmalen nicht in einer Teilmenge vorkommen können, vergleichen wir bei den Experimenten der dritten Dimension nur die ersten drei Methoden.

Wir beschränken uns bei unseren Experimenten auf eine Verteilung, der unsere Merkmale folgen: die multivariate Gaußverteilung. Diese kombinieren wir anschließend mit einem Sampling-Modell, das neben weiteren auch von Aas et al. verwendet wurde [AJL21]. Insgesamt ergibt sich ein Experiment der dritten Dimension, das die drei Modifikationen mit der Kernel-SHAP-Methode und den exakten Shapley-Werten vergleichen.

Zunächst stellen wir nachfolgend die Verteilungen und das Sampling-Modell vor. Wir verwenden bei dem Sampling-Modell eine Störgröße $\varepsilon$. Dieses ist normalverteilt mit $\mathcal{N}(0, 0.01)$.

### 6.2.1. Gaußverteilte Merkmale

Die Input-Daten $X$ folgen der Verteilung $\mathcal{N}_3(\mu, \Sigma(\rho))$, wobei

$$\mu = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{ und } \Sigma = \begin{pmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{pmatrix}. \tag{6.2.1}$$

Der Koeffizient $\rho$ kann dabei zwischen 0 und 0.98 variieren.

### 6.2.2. Lineares Modell

Nachdem wir die Input-Daten für unsere Experimente generiert haben, stellt sich die Frage, wie wir zu unseren Ergebnissen, also unserem Output kommen. Dafür betrachten wir als Erstes ein einfaches lineares Sampling-Modell:

$$y = g(x) = x_1 + x_2 + x_3 + \varepsilon. \tag{6.2.2}$$

Wir erhalten auf diese Weise für jede Zeile unseres Datensatzes $X$ einen Output $y$. Mithilfe dieser Daten können wir nun unser Vorhersagenmodell $g(x)$ anpassen. Wir wählen dafür eine lineare Regression mit den Parametern $\beta_1, \beta_2$ und $\beta_3$.

### 6.2.3. Ergebnisse

Da die Idee unserer Modifizierungen darauf beruht, repräsentativere bzw. realistischere Hintergrunddaten zu generieren, ist für uns nicht nur der MAE und der Skill Score aus den Definitionen 6.1.1 und 6.1.2 relevant: Bevor wir diese betrachten, interessieren wir uns für den erzeugten Hintergrunddatensatz.

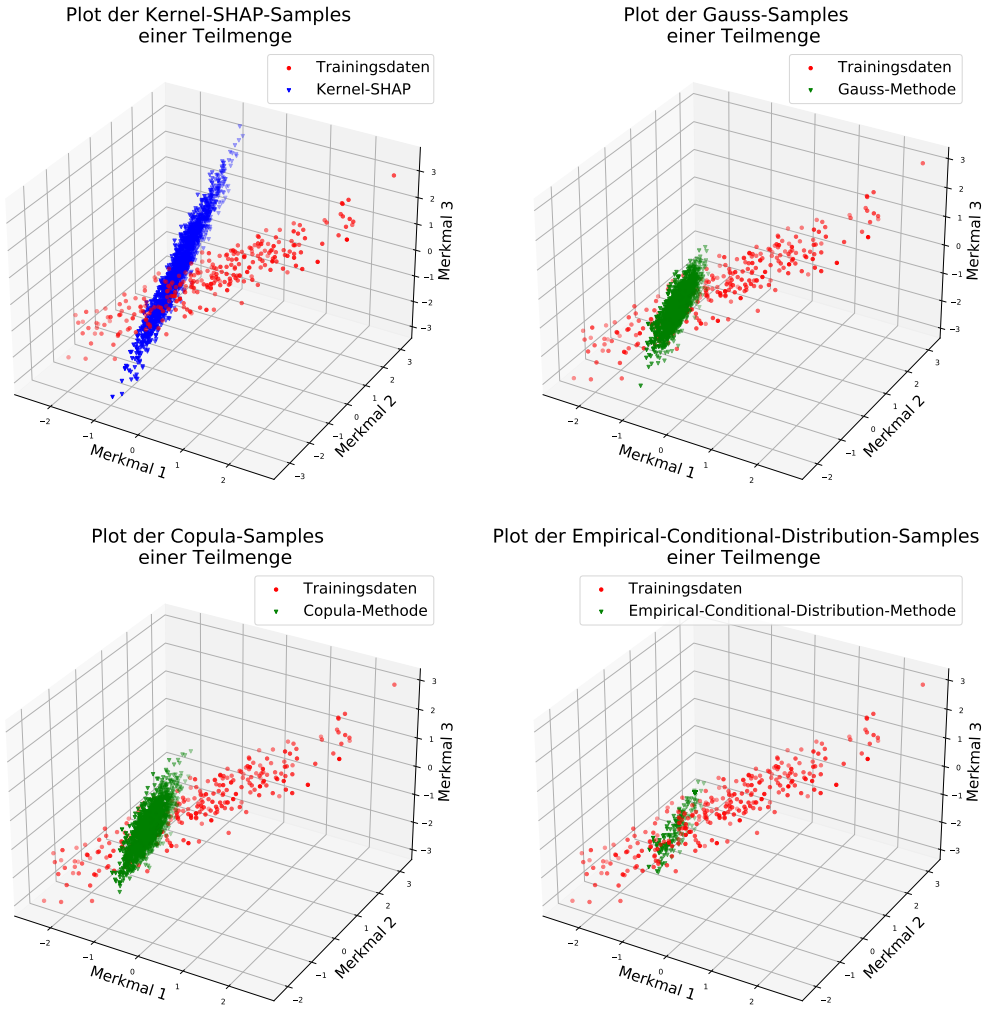



Abbildung 6.1.: **Plots des Trainingsdatensatzes und der Hintergrunddatensätze für die Teilmenge $\mathcal{S} = \{1\}$.** Die roten Punkte beschreiben den gesamten Trainingsdatensatz. Für die einzelnen Methoden wurde je nur die Teilmenge geplottet, für welche Merkmal 1 bei $-0.797898$ fixiert wurde, daher wirken die Stichproben der Methoden wie Ebenen bei $x_1 = -0.797898$. In Blau sieht man die Stichproben der originalen Kernel-SHAP-Methode und in Grün jeweils die Modifizierungen.

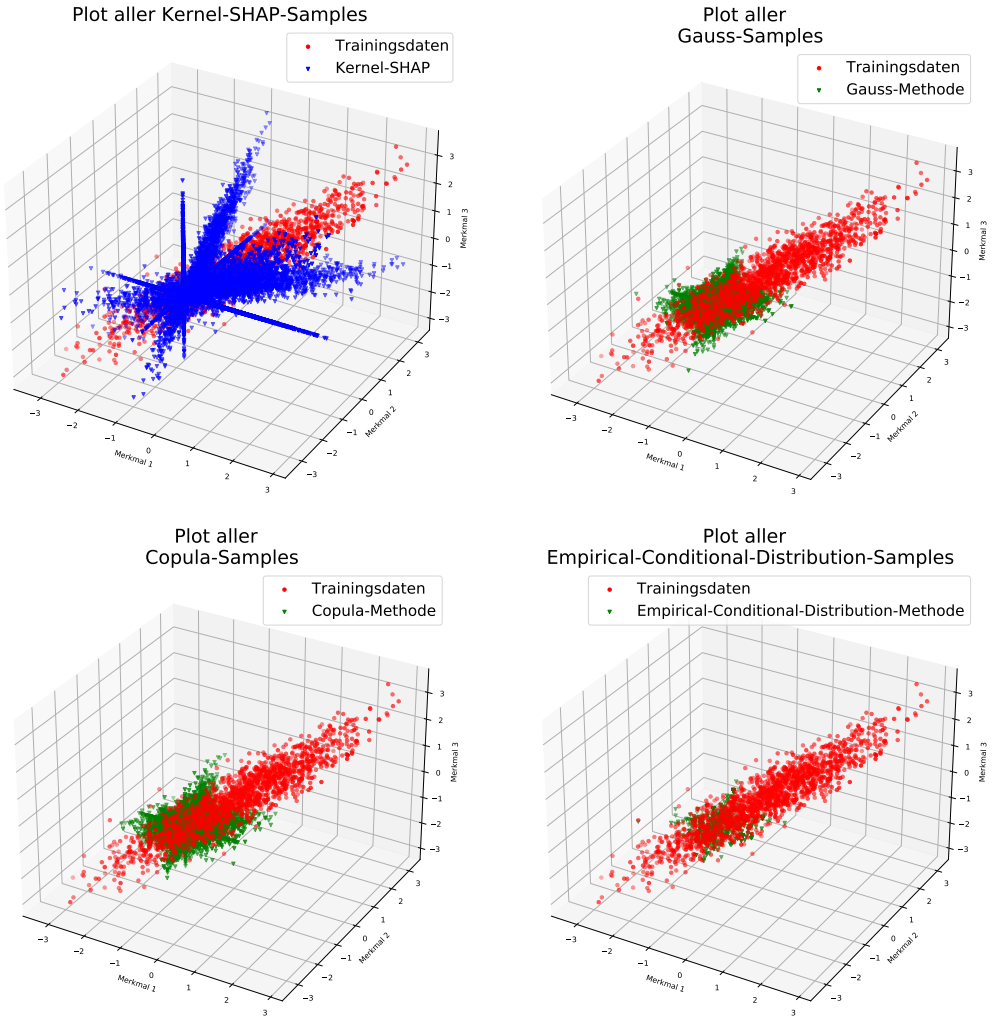



Abbildung 6.2.: **Plot des Trainingsdatensatzes und gesamte Hintergrunddatensätze.** Die roten Punkte beschreiben wieder den gesamten Trainingsdatensatz. Für die einzelnen Methoden wurde jeweils der Hintergrunddatensatz geplottet. In Blau sieht man die Stichproben der originalen Kernel-SHAP-Methode und in Grün jeweils die Modifizierungen.

Wir schauen uns dazu Plots der Hintergrunddaten für eine zufällig gewählte zu erklärenden Instanz $x = [-0.797898, -0.993836, -0.706929]$ an. Zur besseren Visualisierung betrachten wir dabei zu Beginn nur die Teilmenge, bei der das erste Merkmal festgehalten wird (Abbildung 6.1). Es gilt in dieser Abbildung $\mathcal{S} = \{1\}$ und $\bar{\mathcal{S}} = \{2, 3\}$. Analoge Plots für die Teilmengen $\mathcal{S} = \{2, 3\}$, $\bar{\mathcal{S}} = \{1\}$ und $\mathcal{S} = \{3\}$, $\bar{\mathcal{S}} = \{1, 2\}$ befinden sich im Anhang in den Abbildungen B.1 und B.2. Anschließend bilden wir die gleichen Plots für die gesamten Hintergrunddaten (Abbildung 6.2). Es sind demnach alle Teilmengen darin enthalten.

Wir können sehen, dass in allen Abbildungen die Daten der Modifizierungen (grüne Punkte) eher im Bereich des Trainingsdatensatzes (rote Punkte) liegen als die Daten der originalen Kernel-SHAP-Methode (blaue Punkte). Besonders bei Betrachtung der Abbildung 6.2, welche die gesamten Datensätze enthält, erkennen wir, dass die Originalmethode viele Stichproben ausbildet, die weit von den eigentlichen Daten entfernt liegen. Die Stichproben der einzelnen Modifizierungen bilden sich ähnlich einer Wolke im Bereich der Trainingsdaten um die zu erklärende Instanz $x = [-0.797898, -0.993836, -0.706929]$.

Dahingegen bildet die Originalmethode Stichproben aus, die an den fixierten Werten innerhalb der einzelnen Teilmengen von den Trainingsdaten ‚abstrahlen'. Diese äußeren Stichproben von Kernel SHAP sind weitestgehend unrealistisch bzw. spiegeln nicht die durch die Trainingsdaten vorgegebenen Korrelationen wider. Daraus können wir schließen, dass unsere neu eingeführten Modifizierungen für die Trainingsdaten repräsentativere Stichproben generieren als Kernel SHAP.

Der Grund, dass bei den Grafiken der empirisch bedingten Methode nur sehr wenige Hintergrunddaten verzeichnet sind, liegt darin, dass nur die auf Basis der Distanz berechneten Instanzen, die am meisten gewichteten sind, verwendet werden.

Ergänzend dazu möchten wir im Folgenden die in den Definitionen 6.1.1 und 6.1.2 eingeführten Vergleichswerte, den MAE und den Skill Score, untersuchen. Dazu wurden diese für die Shapley-Werte der Testdaten aus unserem Experiment in Abhängigkeit des Korrelationsfaktors $\rho$ berechnet und sind in den nachstehenden Abbildungen 6.3 aufgetragen.

Für annähernd unabhängige Merkmale, also ein sehr kleines $\rho$, liefert die originale Kernel-SHAP-Methode sehr gut Ergebnisse. Sie berechnet also SHAP-Werte, die nah an den ‚wahren' Shapley-Werten liegen. Allerdings ergeben alle drei Modifizierungen – insbesondere für eine höhere Korrelation als ca. $\rho = 0.15$ unter den einzelnen Merkmalen – einen niedrigeren Fehler als die Originalmethode. Das bedeutet, dass jede unserer Modifizierungen SHAP-Werte liefert, die näher an den ‚wahren' Shapley-Werten liegen. Dabei schneidet die Gauß-Modifizierung insgesamt am besten ab.

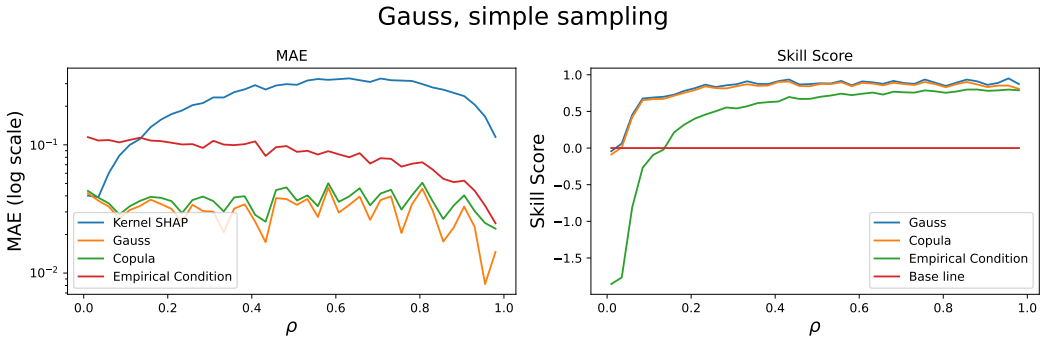



Abbildung 6.3.: **MAE-Plot und Skill Score Plot der Gaußverteilung mit dem linearen Modell in der dritten Dimension.** In Abhängigkeit des Korrelationsparameters $\rho$ der MAE der verschiedenen Methoden auf logarithmierter Skalar und der dazugehörige Skill Score für die gaußverteilten Merkmale in der dritten Dimension.

## 6.3. Experimente mit sechs Merkmalen

Um die vierte und fünfte Methode zu testen, benötigen wir eine höhere Dimension. Die eingeführten Kombinationsmethoden entscheiden basierend auf der Teilmengengröße für jede Teilmenge erneut, ob die empirisch bedingte Methode oder die Gauß- bzw. Copula-Methode angewandt werden soll. Wir betrachten in unseren Experimenten sechs Merkmale, um den Rechenaufwand in vertretbaren Grenzen zu halten, und beschränken uns aus dem gleichen Grund auf das folgende Experiment: Wir kombinieren die gaußverteilten Merkmale mit der gleichen Verteilung wie in der dritten Dimension mit dem nachstehenden linearen Modell.

Die Input-Daten $X$ folgen dementsprechend der multivariaten Normalverteilung $\mathcal{N}_6(\mu, \Sigma(\rho))$, die den Parametern in 6.2.1 in der sechsten Dimension entsprechen.

Das einfache Sampling-Modell, das mit der gewöhnlichen linearen Regression einhergeht, entspricht der Erweiterung von 6.2.2:

$$y = g(x) = \sum_{j=1}^{6} x_j + \varepsilon.$$

Wie auch in den dreidimensionalen Experimenten ist $\varepsilon$ der Störfaktor, welcher der Verteilung $\mathcal{N}(0, 0.01)$ folgt. Die lineare Regression stellt sich analog dar durch:

$$f(x) = \sum_{j=1}^{6} \beta_j x_j + \varepsilon.$$

Auch hier möchten wir die Ergebnisse gerne anhand des MAEs und des Skill Scores der Definitionen 6.1.1 und 6.1.2 validieren. Analog zur dritten Dimension ergibt sich der nachstehende Plot.

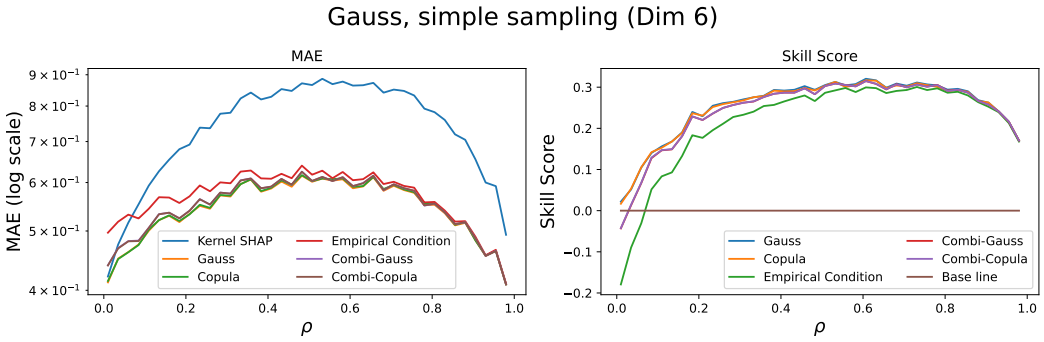



Abbildung 6.4.: **MAE-Plot und Skill Score Plot der Gaußverteilung mit dem linearen Modell in der sechsten Dimension.** In Abhängigkeit des Korrelationsparameters $\rho$ der MAE der verschiedenen Methoden auf logarithmierter Skalar und der dazugehörige Skill Score für die gaußverteilten Merkmale in der sechsten Dimension.

Die Ergebnisse, die das Experiment in der sechsten Dimension liefert, unterscheiden sich nur geringfügig von denen der dritten Dimension. Besonders für eine hohe Korrelation $\rho$ unter den Merkmalen liefern die Modifizierungen ein besseres Ergebnis in dem Sinne, dass sie näher an den ‚wahren' Shapley-Werten als die Originalmethode sind. Wieder ergibt sich, dass die Methode mit der Gauß-Modifizierung den geringsten MAE und damit höchsten Skill Score liefert.

Aas et al. haben in ihrer Validierung zusätzlich die ‚Generalisierte Hyperbolische' Verteilung sowie ein weiteres Sampling-Modell basierend auf stückweise konstanten Funktionen verwendet. Bei ihren Experimenten der zehnten Dimension mit der Generalisierten Hyperbolischen Verteilung ist aufgefallen, dass unsere vierte und fünfte Methode am besten abschneiden. Die Ergebnisse unserer Validierung stimmen mit den Resultaten ihrer etwas umfassenderen Experimente überein [AJL21].

## 6.4. Finanzmodell

Neben der Validierung möchten wir die neu eingeführten Methoden gerne im Rahmen einer realen Anwendung testen: Mithilfe der Shapley-Werte werden wir Vorhersagen eines Finanzmodells, genauer eines Kreditmodells, betrachten.

Zu Beginn machen wir uns mit dem Datensatz vertraut: Welche Merkmale werden betrachtet? Wie sind diese verteilt? Und welche Korrelationen weist der Datensatz auf? Daraufhin untersuchen wir, wie sich unsere neuen Methoden auf den Datensatz auswirken.

### 6.4.1. Beschreibung des Modells

Der Datensatz, den wir verwenden, hat 17 Merkmale. Welche das genau sind und wofür sie stehen, wird in der Tabelle C.1 im Anhang erklärt. Zusammengefasst bildet das Modell eine Vorhersage darüber, ob ein Klient für die Rückzahlung eines Kredites zahlungsfähig ist. Das Modell verwendet als Inputs unter anderem das Verhältnis von Kreditbetrag zu Einkommen, Alter, Bildung, Wohngegend, Geschlecht und Beschäftigungsart und gibt auf Grundlage dieser zurück, ob ein Klient Zahlungsschwierigkeiten haben wird. Die Inputs sind durch numerische, ordinale und kategorische Daten gegeben. Da unsere Modifizierungen nur mit numerischen und ordinalen Inputs arbeiten können, wurden die kategorischen Daten in ordinale umgeschrieben. Wir interessieren uns an dieser Stelle besonders für die Numerischen.

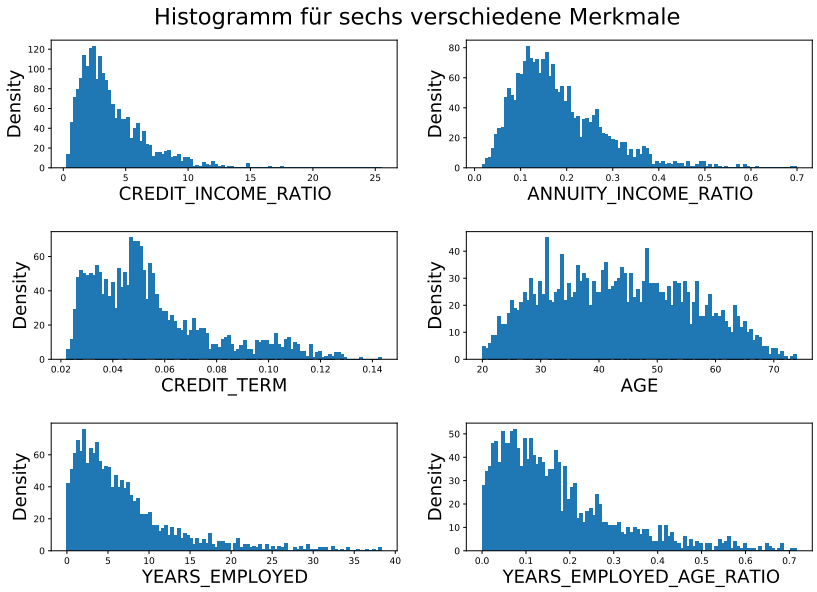



Abbildung 6.5.: **Histogramme verschiedener Merkmale des betrachteten Datensatzes.** Dargestellt sind Häufigkeiten verschiedener Merkmale in Histogrammen. Bei den ersten drei Merkmalen ist eine höhere Korrelation zu erwarten, gleiches gilt für die letzten drei.

Wir verwenden das Modell ‚ExtraTreesClassifier' des ‚sklearn-Package', das 1000 randomisierten Entscheidungsbäumen mit einer maximalen Tiefe von 30 anpasst. Welches wir verwenden, ist allerdings für den eigentlichen Vorgang nicht relevant. Das Modell wurde mit dem gesamten Datensatz trainiert. Da das Arbeiten mit 153.597 Instanzen, die der Datensatz eigentlich beinhaltet, nicht praktikabel ist, gruppieren wir diese mithilfe eines ‚Kmeans-Cluster'-Algorithmus, sodass wir die SHAP-Werte auf Basis von 1000 Datenpunkten bestimmen können. Das bedeutet, dass Punkte, die

nah aneinander liegen, zusammengefasst und zudem mit Gewichten versehen werden, die widerspiegeln, wie viele Daten eine gruppierte Instanz enthält. Daraufhin haben wir diesen neu entstandenen Datensatz, der 1000 Zeilen führt, zusammen mit einer zufällig gewählten Instanz aus unserem Testdatensatz an unsere jeweiligen ‚Explainer' übergeben, um so die verschiedenen SHAP-Werte zu bestimmen.

Bevor wir uns die Korrelationen der Merkmale unseres Datensatzes anschauen, betrachten wir, wie einige Merkmale verteilt sind.

Viele der Merkmale des Modells sind ordinal. Die Verteilung einiger numerischer Inputs können wir in der Abbildung 6.5 erkennen. Man sieht, dass die Merkmale endlastig verteilt sind.

Im Folgenden sehen wir die den Merkmalen zugehörige Korrelationsmatrix. Anhand dieser können wir feststellen, welche der Merkmale stark korreliert sind.

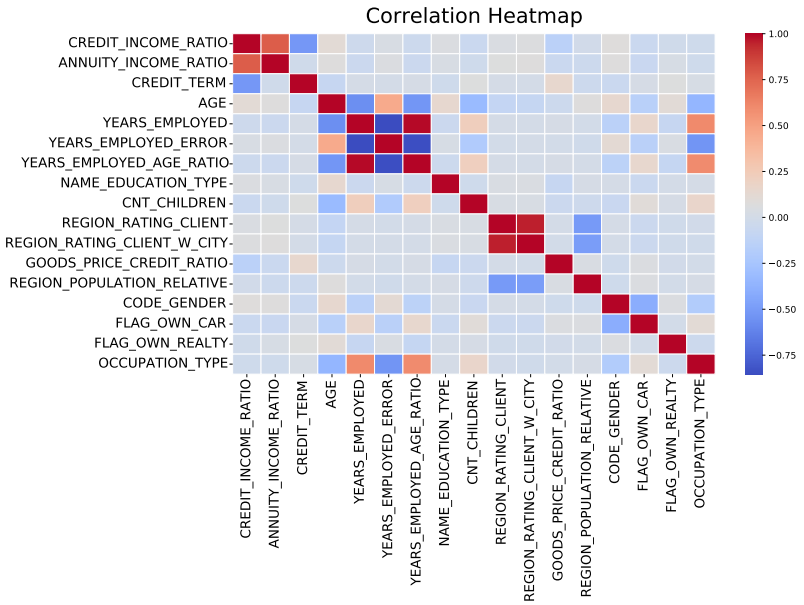



Abbildung 6.6.: **Korrelationsmatrix der Daten im Finanzmodell.** Mithilfe der Farbskala an der rechten Seite kann man dem Diagramm entnehmen, wie stark zwei Merkmale korreliert sind.

Die Merkmale ‚credit_income_ratio' (Verhältnis von Kreditbetrag zu Einkommen) und ‚annuity_income_ratio' (Verhältnis von Annuität zu Einkommen) sind beispielsweise in einem stärkeren Rot gefärbt. Aufgrund dieser intensiven Färbung wissen wir, dass diese beiden Merkmale stark positiv korreliert sind. Im Gegensatz dazu sind ‚credit_income_ratio' und ‚credit_term' (Verhältnis von Kreditbetrag zu Annuität) blau markiert. Sie sind also stark negativ korreliert. Eine positive Korrelation erkennen wir zwischen ‚years_employed' (Länge des bisherigen Arbeitsverhältnisses)

und ‚years_employed_age_ratio ‘ (Verhältnis von Länge des bisherigen Arbeitsverhältnisses zu Alter). Die Merkmale unseres Datensatzes weisen an verschiedenen Stellen Abhängigkeiten auf.

### 6.4.2. Verschiedene Hintergrunddaten

Wie die Unterschiede zwischen der Kernel-SHAP-Methode und unseren Modifizierungen präsentiert werden können, erscheint auf den ersten Blick nicht ganz eindeutig. Anhand der SHAP-Werte, also den Ergebnissen unserer Methoden, sieht man nur sehr wenig. Die genaue Verteilung, der unser Datensatz folgt, ist nicht bekannt. Wie bereits im Kapitel 6.1 der Validierung erwähnt, kommt erschwerend hinzu, dass schlecht einzuordnen ist, welche Methode näher an den ‚wahren‘ Shapley-Werten ist, weil auch bei der Berechnung dieser aktiv eine Funktion ausgewählt werden und die richtige Verteilung bekannt sein muss. Wir benötigen einen anderen Weg, um unsere Modifikationen zu präsentieren.

Wie schon bei der Validierung im vorherigen Kapitel betrachten wir die von den verschiedenen Methoden erzeugten Hintergrunddaten. Um die Methode zu bestimmen, welche die repräsentativeren SHAP-Werte berechnet, vergleichen wir je den Hintergrunddatensatz einer Modifizierung mit dem Hintergrunddatensatz der originalen Kernel-SHAP-Methode und den Trainingsdaten. Wenn der Datensatz der Modifizierung repräsentativer für die Trainingsdaten ist, gehen wir davon aus, dass unsere neue Methode akkurater arbeitet.

An dieser Stelle können wir ebenfalls nicht jede Teilmenge mit allen Merkmalen betrachten: Bei 17 Merkmalen gäbe es $2^{17} = 131.072$ Teilmengen. Stattdessen betrachten wir nur einige Möglichkeiten, die nach dem Wesentlichen gewählt sind: einer hohen Korrelation.

Zwecks besserer Übersicht beschränken wir uns auf die Merkmale ‚credit_income_ratio‘ und ‚annuity_income_ratio‘, die, wie wir anhand der Korrelationsmatrix in Abbildung 6.6 gesehen haben, stark korreliert sind. Für eine aus dem Datensatz zufällig gewählte Instanz erzeugen wir den Trainingsdatensatz sowie die Hintergrunddaten der Copula-Methode und der empirisch bedingten Methode (Abbildung 6.7) und plotten deren Werte für die zwei gewählten Merkmale gegeneinander.

Während in der Abbildung 6.7 im oberen Bild die Originalmethode Punkte entlang ‚annuity_income_ratio‘= 0.1894 ausbildet, ist dies bei Copula nicht der Fall. Die Copula-Methode besteht darin, den Datensatz zu transformieren, dann die Hintergrunddaten zu bestimmen und diese wieder zurück zu transformieren. Dementsprechend setzen wir nicht den ursprünglichen Wert an dieser Stelle fest, sondern die hin und zurück transformierte Version.

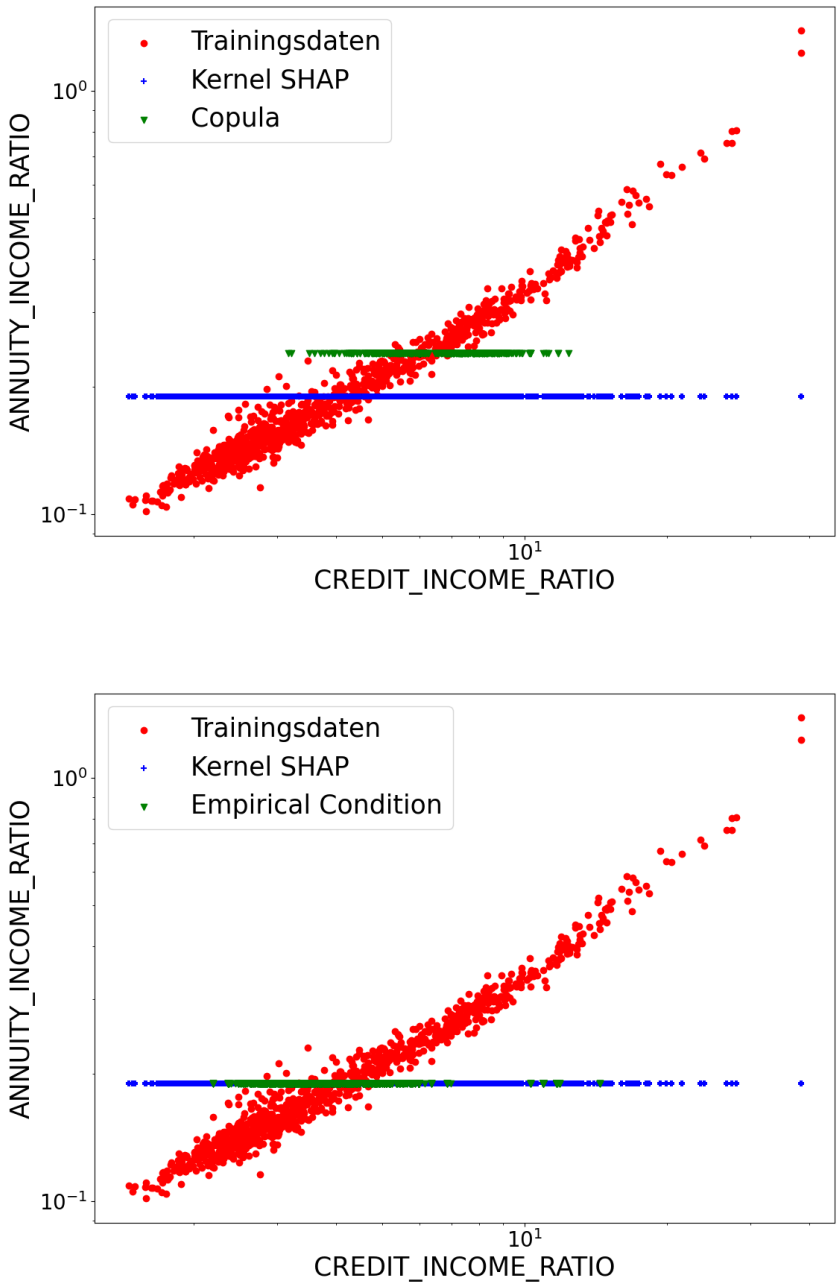



Abbildung 6.7.: **Plot des Trainingsdatensatzes und der Hintergrunddatensätze der ersten beiden Merkmale.** Die roten Punkte beschreiben den gesamten Trainingsdatensatz. Der Plot umfasst alle gruppierten Trainingsdaten und die Instanzen der Hintergrunddaten, die bei ‚annuity_income_ratio ‘= 0.1894 fixiert sind. In blau sieht man die Stichproben der originalen Kernel-SHAP-Methode und in grün jeweils die Modifizierungen.

Wie auch bei der Validierung können wir sehr gut erkennen, dass die Datenpunkte, die von der Originalmethode erzeugt werden, über die eigentlichen Trainingsdaten hinausgehen. Bei unseren Modifikationen ist das ebenfalls geringfügig der Fall, was allerdings darauf zurückzuführen ist, dass hier anstelle des gesamten Trainingsdatensatzes die gruppierte Variante geplottet wurde. Sowohl die Copula-Methode als auch die empirisch bedingte Methode weisen beide Datenpunkte auf, die repräsentativer für die Trainingsdaten sind. Wenn einem Modell nun Daten übergeben werden, die weit von denen entfernt sind, mithilfe derer es trainiert wurde, werden die Vorhersagen unzuverlässiger. Wie bereits erwähnt, ist dies ein Hinweis darauf, dass die Modifikationen akkuratere Ergebnisse liefern.

### 6.4.3. Verschiedene SHAP-Werte

Zum Abschluss möchten wir gerne noch einmal die SHAP-Werte verschiedener Methoden vergleichen. Dazu schauen wir uns zwei verschiedene Plots an, die für zwei Instanzen des Testdatensatzes (also zwei Personen) stehen. Im Folgenden sehen wir in Abbildung 6.8 zwei Balkendiagramme, die für die Originalmethode, die Gauß-Methode und die Copula-Methode die SHAP-Werte gegenüberstellt.

Ohne Weiteres ist sowohl für die erste als auch für die zweite Instanz zu erkennen, dass diese in den verschiedenen Methoden unterschiedlich sind. In dem oberen Diagramm der Abbildung 6.8 können wir vor allem bei den untersten beiden Merkmalen, also ‚credit_income_ratio' und ‚annuity_income_ratio' unterschiedliche SHAP-Werte für die Originalmethode und die beiden Modifikationen feststellen. Bei einem Blick in die Korrelationsmatrix in Abbildung 6.6 sehen wir, dass diese beiden Merkmale sehr stark miteinander korreliert sind. In dem unteren Diagramm kann man das Gleiche für ‚years_employed', ‚age' und ‚years_employed_age_ratio' feststellen. Auch hier liegt laut der Korrelationsmatrix eine hohe Korrelation zwischen ‚years_employed' und ‚years_employed_age_ratio' und eine negative Korrelation dieser beiden Inputs mit ‚age' vor.

Je nach gewählter Methode ergeben sich demnach für die Beiträge der einzelnen Merkmale – also der jeweiligen SHAP-Werte – für denselben Datensatz im gleichen Modell unterschiedliche Ergebnisse.

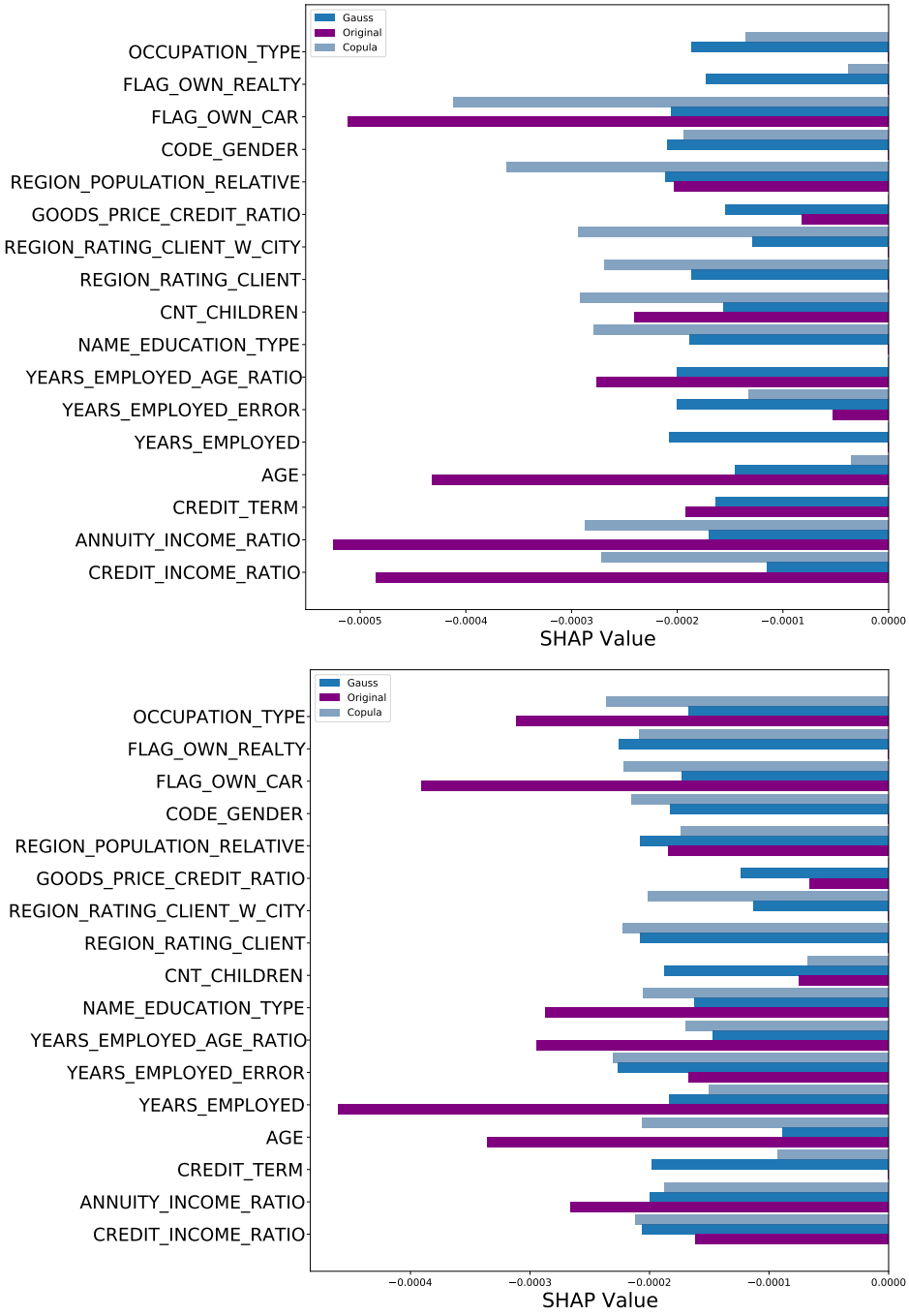



Abbildung 6.8.: **Balkendiagramm der SHAP-Werte verschiedener Methoden für zwei Instanzen.** Für die einzelnen Merkmale sind die SHAP-Werte der originalen Kernel-SHAP-Methode, der Gauß- und der Copula-Methode dargestellt. Die obere Abbildung stellt dabei eine Person aus dem Datensatz dar, die untere eine weitere.

# 7. Fazit

Der wesentliche Kern dieser Arbeit war es, die Kernel-SHAP-Methode hinsichtlich Abhängigkeiten ihrer Inputs mit verschiedenen Methoden zu erweitern und diese zu testen.

## 7.1. Ergebnisse

Zu Beginn haben wir Shapley-Werte kennengelernt und uns mit nennenswerten Eigenschaften und ihrer Anwendung im ursprünglichen Sinne, in der koalitiven Spieltheorie, auseinandergesetzt. Nachdem ihr zugrundeliegendes Prinzip verdeutlicht wurde, haben wir Parallelen zu der Erklärung von Black-Box-Modellen geschaffen und uns auf diesem Weg mit SHAP-Werten und ihrer Funktionsweise vertraut gemacht.

Im Bereich der SHAP-Werte haben wir uns auf ein modell-agnostisches Verfahren, Kernel SHAP, konzentriert und dessen Arbeitsweise sowohl von der mathematischen Seite als auch auf die im Code implementierte Art und Weise betrachtet. Dabei konnten wir feststellen, dass eine wichtige Annahme getroffen wird, die einen weitreichenden Einfluss auf die Qualität der Ergebnisse haben könnte: Wir approximieren einen Erwartungswert, indem wir die bedingte Dichtefunktion durch ihre nicht-bedingte Form ersetzen – wir nehmen also Unabhängigkeit zwischen den Merkmalen an.

Dieses Wissen haben wir in Kapitel 5 verwendet, um vier verschiedene Methoden kennenzulernen, die diese Annahme nicht verwenden. Auf der Basis einer multivariaten Gaußverteilung, einer Gauß-Copula oder einem neu entwickelten Verfahren, das die jeweiligen Randverteilungen miteinbezieht, wurden Modifikationen an der Kernel-SHAP-Methode durchgeführt, die Korrelationen zwischen den Inputdaten beachten und damit die bedingte Dichte korrekt verwenden können. An dieser Stelle haben wir in Python die bereits bestehende ‚KernelExplainer'-Klasse des ‚SHAP'-Pakets um diverse Unterklassen ergänzt, die diese neuen Modifikationen in das Framework integrieren und somit eine sinnvolle Erweiterung dieses darstellen. Wir haben ihre genaue mathematische Umsetzung sowie die neue Implementierung besprochen.

Diese neuen Methoden haben wir im Rahmen einiger Experimente verschiedener Dimensionen validiert. Abschließend haben wir sie zusätzlich im Zusammenhang mit einem Finanzmodell anhand eines realen Datensatzes zur Kreditausfallschätzung getestet.

In der Validierung konnten wir feststellen, dass die neu eingeführten Methoden näher an den Shapley-Werten liegen, wenn diese auf Basis des bedingten Erwartungswertes der Daten berechnet wurden. Das Kreditmodell hat verdeutlicht, dass der Einsatz verschiedener Methoden unterschiedliche SHAP-Werte erzeugt.

## 7.2. Ansätze zur zukünftigen Weiterentwicklung

Auf den ersten Blick wirken Shapley-Werte in der Anwendung bei Erklärungsmodellen wie ein ausgesprochen hilfreicher Ansatz: Für den Nutzer erscheint es intuitiv, die SHAP-Werte als den jeweiligen Beitrag eines Merkmals zu einer Vorhersage zu interpretieren. Universell einsetzbar erleichtern sie auf einer erprobten, mathematisch fundierten Grundlage das Deuten von Blackbox-Modellen.
In der Folge werden wir jedoch dazu verleitet, diese nicht mehr zu hinterfragen. Diese bedingungslose Hinnahme ist problematisch, da der Verwendung der SHAP-Werte die Wahl der Funktion $f_x$ zugrunde liegt.

Das Beispiel 1.0.1 zu Beginn dieser Arbeit bestand darin, Wohnungspreise auf Grundlage verschiedener Merkmale zu schätzen. Wenn wir nun nicht nur unsere originale Kernel-SHAP-Methode, sondern zusätzlich unsere neuen Methoden darauf anwenden, könnten andere Ergebnisse für die einzelnen Beiträge der Merkmale erzeugt werden. Insbesondere bei den Merkmalen, die eine hohe Korrelation haben – wie z. B. ‚Anzahl der Zimmer' und ‚Quadratmeteranzahl' – sind andere SHAP-Werte zu erwarten, da bei unseren neuen Methoden Abhängigkeiten berücksichtigt werden.

Mithilfe dieser Modifikationen wird den Nutzern demnach noch einmal deutlicher vor Augen geführt, dass es in der Erklärung von Vorhersagen auf Basis realer Daten nicht nur ein Ergebnis gibt. Je nach gewählter Näherung bzw. gesetzten Annahmen werden unterscheidende Resultate für die Beiträge erzeugt. Hilfreich sind SHAP-Werte also in jedem Fall weiterhin, es lohnt sich aber, diese zu vergleichen und sich intensiver mit ihnen auseinanderzusetzen, was unsere Modifikationen möglich machen.

Insgesamt bieten die Kernel-SHAP-Methode wie auch ihre von uns eingebrachten Ergänzungen viele Weiterentwicklungsmöglichkeiten, die in zukünftigen Studien erprobt werden können.
Die neuen Modifizierungen bieten die Möglichkeit, die bedingte Dichte statt ihrer nicht-bedingten Approximation zu verwenden. Im Vergleich zur Kernel-SHAP-Methode lassen wir demnach eine unrealistische Annahme – die Unabhängigkeit der Merkmale – weg. Zugleich bedeutet dies, dass unsere neuen Methoden komplexer sind, womit der Rechenaufwand ansteigt. Chen et al. haben versucht, die dem Datensatz zugrundeliegende Graphenstruktur zu nutzen, um die Rechenkomplexität zu minimieren [Che+18]. Möglicherweise kann dies mit einbezogen werden und in Zukunft zu schnelleren Ausführungen des Programms verhelfen.

Aufgrund dieser hohen Rechenkomplexität erscheint ein ebenfalls sinnvoller Fokus die Bemühung um eine effizientere Implementierung auf Basis numerischer Approximationen. Es könnten Studien zum Approximationsfehler der Erwartungswerte von Originalmethoden betrieben werden – möglicherweise im Vergleich zu den Modifizierungen oder in Abhängigkeit gegebener Parameter wie der Stichprobenanzahl.

Ein zukünftiger Ausbau könnte in der Erweiterung auf kategorische Daten bestehen. In unserem Experiment des Kreditmodells wurden die Daten vorher so modifiziert, dass unsere Methoden darauf Anwendung finden konnten. Eine allgemeine Version, die universell anwendbar ist, wäre sicherlich von Vorteil. Von Aas et al. wird dafür ‚One-Hot-Encoding' oder eine erweiterte Form des ‚Cluster-Algorithmus' nach Zhexue Huang [Hua97] vorgeschlagen [AJL21].

Obwohl unsere Methode zurzeit auf tabellarische Daten beschränkt ist, wäre auch hier eine zusätzliche Ergänzung möglich: Im Allgemeinen können auch Texte oder Bilder durch die Kernel-SHAP-Methode erklärt werden. Ebenfalls können unsere Modifikationen durch die Berechnung der Parameter $\sigma$ und $\eta$ in der dritten Methode ausgebaut werden, die allerdings wiederum einen höheren Rechenaufwand mit sich brächte.

Insgesamt bilden die Modifizierungen der Kernel-SHAP-Methode als valide Alternative zur Originalen einen guten Anfang zur weiteren Forschung. Sie bilden für reale Datensätze eine Möglichkeit, Korrelationen zu berücksichtigen, was zu realistischeren Stichproben führt, die zur Berechnung der Beiträge der Merkmale benötigt werden. Darüber hinaus bieten sie das Untersuchen und Vergleichen der verschiedenen Shapley-Werte auf Basis der Wahl der Funktion $f_x$ an, die das Auslassen der Merkmale simuliert.

# A. Vorgehen bei Kernel SHAP

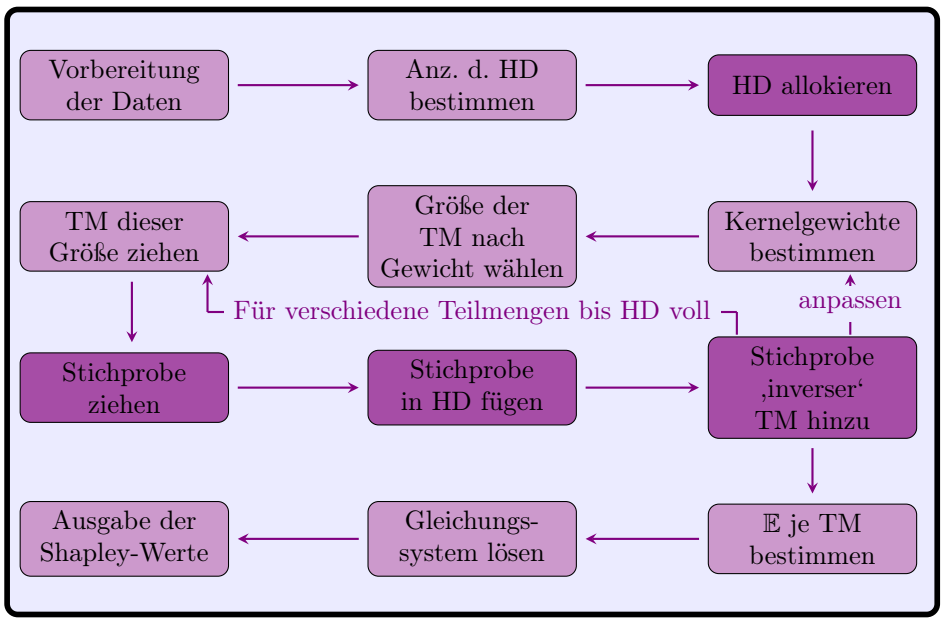



Abbildung A.1.: **Ablaufdiagramm des Vorgehens im Code.** In diesem Diagramm wird das allgemeine, stark vereinfachte Vorgehen im Code veranschaulicht. ‚TM' steht hierbei für Teilmenge und ‚HD' für Hintergrunddaten. Die dunkler gefärbten Schritte müssen größenteils im Rahmen der Modifizierungen geändert werden.

# B. Weitere Plots der Validierung

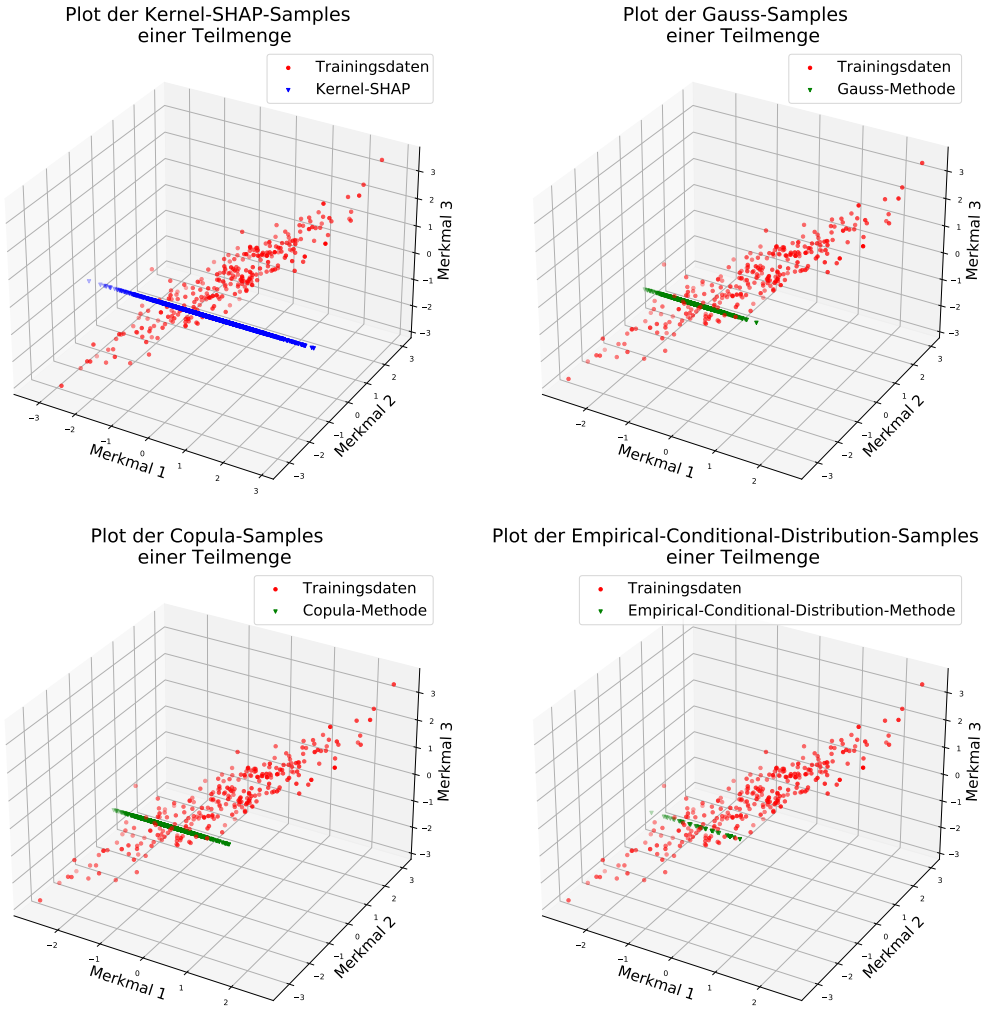



Abbildung B.1.: **Plot des Trainingsdatensatzes und der Hintergrunddatensätze für die Teilmenge $\mathcal{S} = \{2, 3\}$ geplottet.** Die roten Punkte beschreiben den gesamten Trainingsdatensatz, die Blauen die Stichprobe der Originalmethode und die Grünen jeweils die der Modifizierungen. Das Merkmal 2 ist bei $-0.993836$ und das Merkmal 3 bei $-0.706929$ fixiert

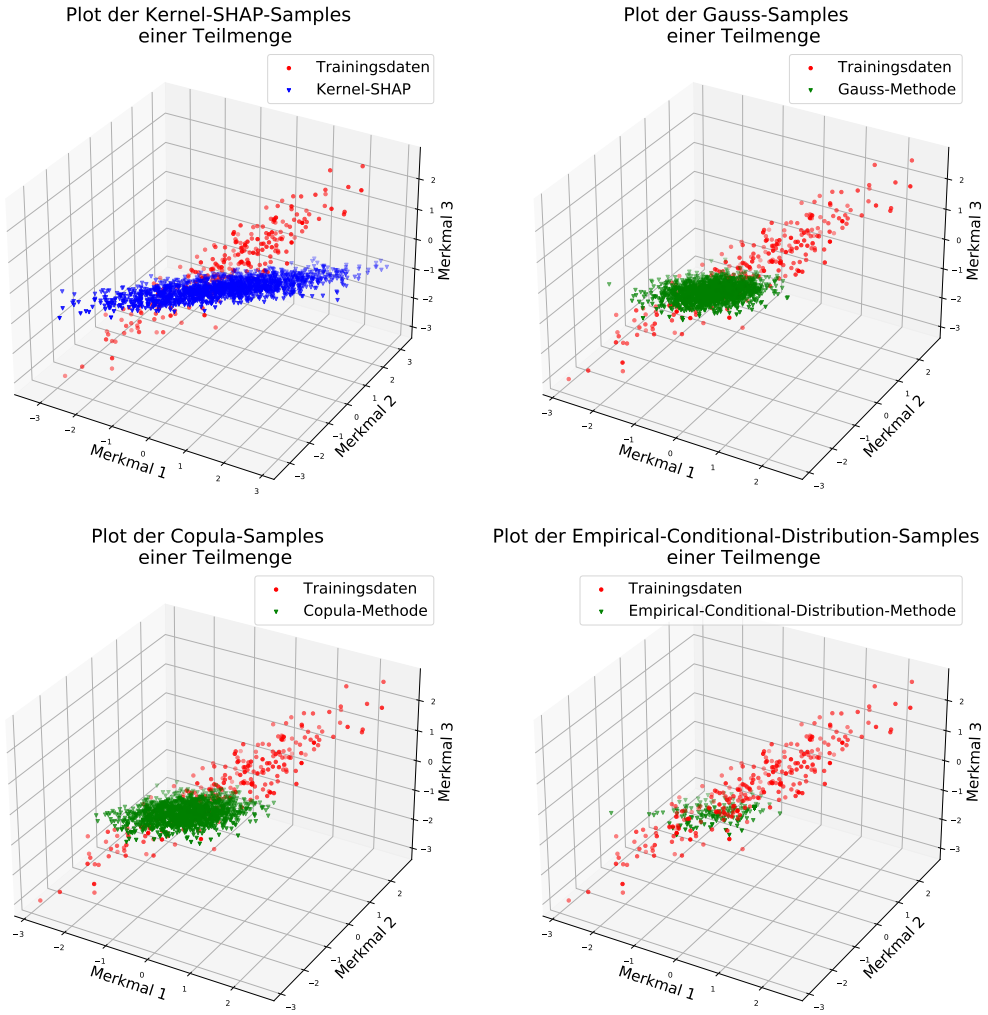



Abbildung B.2.: **Plot des Trainingsdatensatzes und der Hintergrunddatensätze für die Teilmenge $\mathcal{S} = \{3\}$.** Die roten Punkte beschreiben den gesamten Trainingsdatensatz. Für die einzelnen Methoden wurde je nur die Teilmenge geplottet, für die Merkmal 3 bei ca. $-0.706929$ fixiert wurde, daher wirken die Stichproben der Methoden wie Ebenen bei $x_3 = -0.706929$. In Blau sieht man die Stichproben der originalen Kernel-SHAP-Methode und in Grün jeweils die Modifizierungen.

# C. Merkmale des Finanzmodells

| Name | Erklärung oder Zusammensetzung |
|---|---|
| amt_credit | Entgültiger Kreditbetrag |
| amt_income_total | Einkommen des Klienten |
| credit_income_ratio | $\frac{\text{amt\_credit}}{\text{amt\_income\_total}}$ |
| amt_annuity | Höhe der Annuitätenzahlung |
| annuity_income_ratio | $\frac{\text{amt\_annuity}}{\text{amt\_income\_total}}$ |
| credit_term | $\frac{\text{amt\_annuity}}{\text{amt\_credit}}$ |
| age | Alter des Klienten |
| years_employed | Länge des bisherigen Arbeitsverhältnisses |
| years_employed_error | years_employed $< 0$ |
| years_employed_age_ratio | $\frac{\text{years\_employed}}{\text{age}}$ |
| name_education_type | Höchster Bildungsabschluss |
| cnt_children | Anzahl der Kinder |
| region_rating_client | Einstufung der Wohnregion |
| region_rating_client_w_city | Einstufung der Wohnregion (inkl. Stadt) |
| amt_goods_price | Warenpreis (Verbraucherkredite) |
| goods_price_credit_ratio | $\frac{\text{amt\_goods\_price}}{\text{amt\_credit}}$ |
| region_population_relative | Normalisierte Bevölkerungsanzahl der Wohnregion |
| code_gender | Geschlecht des Klienten |
| flag_own_car | Besitz eines Autos |
| flag_own_realty | Besitz von Immobilien |
| occupation_type | Art der Anstellung |
| target | Output des Modells: Zahlungsschwierigkeiten |

Abbildung C.1.: **Beschreibung der Modellmerkmale.** Auf der linken Seite sieht man die Namen der Merkmale und auf der rechten Seite ist die Berechnung zu erkennen, falls diese sich aus weiteren zusammensetzen, oder die Beschreibung dessen, wofür sie stehen. Eine Zeile ist in grau markiert, wenn sie aktiv im Modell nicht angewandt wird, sondern nur indirekt verwendet wird.

# Abbildungsverzeichnis

# Literatur


[AJL21] Kjersti Aas, Martin Jullum und Anders Løland. "Explaining individual predictions when features are dependent: More accurate approximations to Shapley values". In: *Artificial Intelligence* 298 (2021).

[Cha+88] A Charnes u. a. "Extremal principle solutions of games in characteristic function form: core, Chebychev and Shapley value generalizations". In: *Econometrics of planning and efficiency*. Springer, 1988, S. 123–133.

[Che+18] Jianbo Chen u. a. "L-shapley and c-shapley: Efficient model interpretation for structured data". In: *arXiv preprint arXiv:1808.02610* (2018).

[Che+20] Hugh Chen u. a. "True to the Model or True to the Data?" In: *arXiv preprint arXiv:2006.16234* (2020).

[CLL20] Ian Covert, Scott M Lundberg und Su-In Lee. "Understanding global feature contributions with additive importance measures". In: *Advances in Neural Information Processing Systems* 33 (2020).

[ELM01] Paul Embrechts, Filip Lindskog und Alexander McNeil. "Modelling dependence with copulas". In: *Rapport technique, Département de mathématiques, Institut Fédéral de Technologie de Zurich, Zurich* 14 (2001), S. 1–50.

[FRF20] Christopher Frye, Colin Rowat und Ilya Feige. "Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability". In: *Advances in Neural Information Processing Systems* 33 (2020), S. 1229–1239.

[FV98] Edward W Frees und Emiliano A Valdez. "Understanding relationships using copulas". In: *North American actuarial journal* 2.1 (1998), S. 1–25.

[GR07] Tilmann Gneiting und Adrian E Raftery. "Strictly proper scoring rules, prediction, and estimation". In: *Journal of the American statistical Association* 102.477 (2007), S. 359–378.

[HST98] Clifford M Hurvich, Jeffrey S Simonoff und Chih-Ling Tsai. "Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60.2 (1998), S. 271–293.

[Hua97] Zhexue Huang. "A fast clustering algorithm to cluster very large categorical data sets in data mining." In: *Dmkd* 3.8 (1997), S. 34–39.

[IL17] Rafael Izbicki und Ann B. Lee. "Converting high-dimensional regression to high-dimensional conditional density estimation". In: *Electronic Journal of Statistics* 11.2 (2017). Publisher: Institute of Mathematical Statistics and Bernoulli Society. DOI: 10.1214/17-EJS1302.

[Kva+18] Håvard Kvamme u. a. "Predicting mortgage default using convolutional neural networks". In: *Expert Systems with Applications* 102 (2018), S. 207–217.

[LL17] Scott M Lundberg und Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in neural information processing systems* 30 (2017).

[Lun+18] Scott M Lundberg u. a. "Explainable machine-learning predictions for the prevention of hypoxaemia during surgery". In: *Nature Biomedical Engineering* 2.10 (2018), S. 749.

[Lun18] Scott Lundberg. *„SHAP documentation!"*. Website. Online erhältlich unter `https://shap.readthedocs.io/en/latest/index.html`; (Besucht am 28. 06. 2022). 2018.

[Lun22] Scott Lundberg. *„SHAP Package"*. Website. Online erhältlich unter `https://github.com/slundberg/shap`; (Besucht am 28. 06. 2022). 2022.

[Mah36] Prasanta Chandra Mahalanobis. "On the generalized distance in statistics". In: National Institute of Science of India. 1936.

[Mol20] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[Nel07] Roger B Nelsen. *An introduction to copulas*. Springer Science & Business Media, 2007.

[RW06] Carl Edward Rasmussen und Christopher KI Williams. *Gaussian processes for machine learning. Number ISBN 0-262-18253-X*. 2006.

[Sha53] Lloyd S Shapley. "Contributions to the Theory of Games, Chapter A Value for n-person Games". In: (1953).

[ŠK10] Erik Štrumbelj und Igor Kononenko. "An efficient explanation of individual classifications using game theory". In: *The Journal of Machine Learning Research* 11 (2010), S. 1–18.

[ŠK14] Erik Štrumbelj und Igor Kononenko. "Explaining prediction models and individual predictions with feature contributions". en. In: *Knowledge and Information Systems* 41.3 (Dez. 2014), S. 647–665. DOI: 10.1007/s10115-013-0679-x. (Besucht am 26. 06. 2022).

[Skl59] M Sklar. "Fonctions de repartition an dimensions et leurs marges". In: *Publ. inst. statist. univ. Paris* 8 (1959), S. 229–231.

[Sud+10] Agus Sudjianto u. a. "Statistical methods for fighting financial crimes". In: *Technometrics* 52.1 (2010), S. 5–19.

[ZRS22] Yilun Zhou, Marco Tulio Ribeiro und Julie Shah. "ExSum: From Local Explanations to Model Understanding". In: *arXiv preprint arXiv:2205.00130* (2022).

# Implementierung