h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

**Hochschule Darmstadt**

**Fachbereich Mathematik**
**und Naturwissenschaften**
**& Informatik**

# Identifikation und Visualisierung von Interaktionen in dünnbesetzten additiven Modellen - Eine Anwendung auf Überlebensdaten nach Nierentransplantation

Abschlussarbeit zur Erlangung des akademischen Grades
Master of Science (M. Sc.)
im Studiengang Data Science

vorgelegt von

Fabian Rasch

Referent: Prof. Dr. Antje Jahn
Korreferent: Prof. Dr. Gunter Grieser

Ausgabedatum: 27. November 2020
Abgabedatum: 14. Mai 2021

# Eidesstattliche Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 10.05.2021

Fabian Rasch

# Datennutzungsvereinbarung

The data reported here have been supplied by the United Network for Organ Sharing as the contractor for the Organ Procurement and Transplantation Network. The interpretation and reporting of these data are the responsibility of the author and in no way should be seen as an official policy of or interpretation by the OPTN or the U.S. Government. The entire analysis is based on OPTN data as of June 20, 2020.

## Kurzfassung

Die Erklärbarkeit von Prognosemodellen spielt insbesondere bei Entscheidungen, die das Leben eines Menschen gravierend beeinflussen können, eine wesentliche Rolle. Additive Regressionsmodelle bieten die Möglichkeit, gut erklärbare Modelle zu erstellen. Durch eine Vielzahl von Erweiterungen lässt sich die Basisvariante an verschiedene Anwendungsfälle anpassen.

Ziel dieser Thesis ist es verschiedene additive Modelle mit Lasso Regularisierung zu recherchieren, um Variablen und Interaktionen zu einer ausgewählten Variable zu selektieren. Im Theorieteil wurden dazu die mathematischen Unterschiede der Verfahren GGLasso, GLINTERNET, GAMSEL und Pliable untersucht.

In einem praktischen Anwendungsfall wurden die vier Methoden auf Nierentransplantationsdaten aus den USA angewandt. Der Kidney Donor Profile Index (KDPI), der für die Nierenqualität steht, spielt eine zentrale Rolle im Transplantationsprozess, daher wurden insbesondere Interaktionen zwischen Merkmalen des Empfängers und dem KDPI untersucht. Hierzu wurde mittels logistischer Regression der Überlebensstatus nach einem Jahr prognostiziert. Durch die Anwendung der oben genannten Methoden, wurden relevante Variablen und Interaktionen identifiziert. Das Modell konnte die Anzahl an Variablen deutlich reduzieren und gleichzeitig die Prognosegüte gegenüber eines Modells ohne Variablenselektion verbessern. Die Erkenntnisse, insbesondere über den Einfluss von Variablen und Interaktionen, wurden in einem interaktiven Dashboard dargestellt.

Um die Methoden systematisch zu vergleichen, wurde zusätzlich eine Simulationsstudie durchgeführt. Hierbei sollten die Verfahren einflussreiche Variablen und Interaktionen unter einer Vielzahl von Störvariablen identifizieren. Mittels dieser Studie und den Ergebnissen aus dem praktischen Anwendungsfall, konnten wesentliche Unterschiede und Probleme in den Verfahren festgestellt werden.

***Schlagworte*** — Nierentransplantation; Additive Modelle; Pliable; GAMSEL; GLINTERNET; GGLasso.

## Abstract

The explainability of prediction models plays an important role, especially in cases where decisions have severe consequences on a person's life. Additive regression models provide an opportunity to create explainable models. The basis version can be adjusted to various use cases by numerous expansions.

The objective of this thesis is to research several additive models with lasso regularization to select variables and interactions with a chosen variable. For that in the theoretical part the mathematical differences between the methods GGLasso, GLINTERNET, GAMSEL and Pliable have been examined.

In a practical use case the four methods have been applied on kidney transplant data from the USA. The Kidney Donor Profile Index (KDPI), which represents the kidney quality, plays a major role in the transplantation process, therefore especially the interactions between attributes of the recipient and the KDPI have been examined. For this the survival outcome of the recipient has been predicted by the use of logistic regression models. Relevant variables and interactions were identified by applying the aforementioned methods. The final model had significant less variables and at the same time increased the predictive accuracy compared to a model without variable selection. An interactive dashboard has been created to show the findings, especially the effects of the selected variables and interactions.

To compare the methods systematically a simulation study has been conducted additionally. The methods had to identify important variables and interactions among numerous noise variables. By means of this study and the results of the practical use case significant differences and problems of the methods have been discovered.

***Keywords*** — kidney transplant; additive models; Pliable; GAMSEL; GLINTERNET; GGLasso.

# Inhaltsverzeichnis

**Abbildungs-/Tabellenverzeichnis** **VII**

**Programmcodeverzeichnis** **IX**

**Abkürzungsverzeichnis** **X**

**1 Einleitung** **1**
1.1 Motivation . . . 1
1.2 Ziel der Arbeit . . . 3
1.3 Gliederung . . . 4

**2 Theoretische Grundlagen** **5**
2.1 Die logistische Regression . . . 5
2.2 Bias-Variance Tradeoff . . . 7
2.3 Die Shrinkage Methoden . . . 8
2.3.1 Die Lasso Regression . . . 9
2.3.2 Die Ridge Regression . . . 9
2.4 Die Kreuzvalidierung . . . 10
2.5 Der Bootstrap . . . 11
2.6 Interaktionen zwischen Variablen . . . 12
2.7 Probleme mit Confounding . . . 13
2.7.1 Der Propensity Score . . . 15
2.7.2 Generalized Propensity Score . . . 16

**3 Spezielle Verfahren zur Variablenselektion** **19**
3.1 Group Lasso . . . 19
3.2 GLINTERNET . . . 20
3.3 Pliable . . . 23
3.4 GAMSEL . . . 25

**4 Anwendung auf Nierentransplantationsdaten** **28**
4.1 Vorgehen . . . 28
4.2 Business Understanding . . . 29
4.2.1 Der Allokationsprozess . . . 30
4.2.2 Der Anwendungsfall . . . 31
4.3 Data Understanding . . . 32
4.3.1 Herkunft und Struktur der Daten . . . 32
4.3.2 Identifikation relevanter Tabellen . . . 33

4.3.3 Erste Einblicke in die Daten . . . 35
4.4 Data Preperation . . . 36
4.4.1 Einschränkung des Patientenkollektivs . . . 37
4.4.2 Variablenvorauswahl . . . 38
4.4.3 Umgang mit Confounding . . . 38
4.4.4 Ergebnis der Datenvorbereitung . . . 40
4.5 Modelling . . . 42
4.5.1 Vorgehen der Modellierung . . . 42
4.5.2 Bestimmung des besten Modells . . . 43
4.6 Evaluation . . . 50
4.7 Deployment . . . 52
4.7.1 Deskriptive Statistik . . . 53
4.7.2 Vorhersagemodell . . . 54
4.7.3 Interaktionen mit KDPI . . . 56

**5 Systematischer Vergleich der Verfahren 57**
5.1 Unterschiede im Anwendungsfall . . . 57
5.2 Simulationsstudie . . . 62
5.2.1 Aufbau der Simulationsstudie . . . 62
5.2.2 Ergebnisse . . . 64

**6 Fazit 68**
6.1 Fazit . . . 68
6.2 Ausblick . . . 69

**Quellenverzeichnis 71**

**Anhang 75**
A Pakete . . . 75
B Programmcode . . . 76

# Abbildungs-/Tabellenverzeichnis

## Abbildungen


1.1 Entwicklung von Nierentransplantationen . . . . . 1
1.2 Einfluss von KDPI & EPTS . . . . . 2
2.1 Bias-Variance Tradeoff . . . . . 8
2.2 Vergleich Lasso & Ridge . . . . . 10
2.3 Kreuzvalidierung . . . . . 11
2.4 Positive Interaktion . . . . . 13
2.5 Confounding . . . . . 14
4.1 CRISP . . . . . 28
4.2 KAS . . . . . 32
4.3 Datenschema . . . . . 33
4.4 Verteilung fehlender Werte . . . . . 35
4.5 Confounding durch Allokationsprozess . . . . . 36
4.6 Verteilung von Alter für KDPI Stufen . . . . . 36
4.7 Die genaue Berechnung des Estimated Post Transplant Survival (EPTS) 39
4.8 Verteilung von Alter für KDPI Stufen gewichtet mit GPS . . . . . 40
4.9 Überschnitt . . . . . 45
4.10 Nicht-lineare Effekte von GAMSEL . . . . . 46
4.11 Modelle für die Kreuzvalidierung . . . . . 47
4.12 Kreuzvalidierungsfehler der Variablenselektion für K=5 . . . . . 48
4.13 Kreuzvalidierungsfehler der Variablenselektion für K=10 . . . . . 49
4.14 Sidebar . . . . . 52
4.15 Deskriptive Statistik . . . . . 53
4.16 Vorhersagemodell . . . . . 54
4.17 Interaktionen mit KDPI . . . . . 56
5.1 Haupteffekte . . . . . 58
5.2 Interaktionseffekte . . . . . 60
5.3 Simulation Teil 1 . . . . . 65
5.4 Simulation Teil 2 . . . . . 66

# Tabellen

3.1 Zusammenfassung über spezielle Verfahren . . . . . 27
4.1 Variablenvorauswahl . . . . . 41
4.2 Beispiel Resultat Tabelle GLINTERNET . . . . . 44
4.3 Modellevaluation . . . . . 51
5.1 Laufzeit der Verfahren . . . . . 62
5.2 Parameterauswahl bei der Simulation . . . . . 63

# Programmcodeverzeichnis

A.1 R Pakete . . . . . 75
B.1 Bootstrap GGLasso . . . . . 76
B.2 Bootstrap GLINTERNET . . . . . 77
B.3 Bootstrap Pliable . . . . . 79
B.4 Bootstrap GAMSEL . . . . . 81
B.5 Simulationstudie . . . . . 82

# Abkürzungsverzeichnis

**KAS** New Kidney Allocation System
**KDPI** Kidney Donor Profile Index
**EPTS** Estimated Post Transplant Survival
**BS** Brier Score
**BSS** Brier Skill Score
**Lasso** Least Absolute Shrinkage and Selection Operator
**u. d. N.** unter der Nebenbedingung
**PS** Propensity Score
**GPS** Generalized Propensity Score
**RCT** Randomized Controlled Trial
**GLINTERNET** Learning interactions through hierarchical group-lasso regularization
**GAMSEL** Generalized Additive Model Selection
**CRISP** Cross Industry Standard Process
**OPTN** Organ Procurement and Transplantation Network
**HLA** Humane Leukozytenantigene
**GFR** glomeruläre Filtrationsrate
**UNOS** United Network for Organ Sharing
**GFR** Estimated Glomerular Filtration Rate

# 1 Einleitung

## 1.1 Motivation

Die Nieren haben eine zentrale Rolle im menschlichen Körper. Sie dienen unter anderem zur Regulierung des Blutdrucks, der Ausscheidung von giftigen Substanzen und der Regulierung des Elektrolytehaushalts. Gesunde Nieren filtern dabei das gesamte Blut bis zu 300 Mal am Tag (Mai 2016). Die Niere kann durch unterschiedliche Krankheiten in ihrer Funktion eingeschränkt oder dauerhaft geschädigt werden. Die häufigsten Ursachen sind hierbei Diabetes und Bluthochdruck, aber auch Entzündungen oder Gendefekte können Nieren stark beeinträchtigen. Versagen die Nieren gänzlich, müssen die Aufgaben künstlich durch die Dialyse übernommen werden (*Die Nierentransplantation* o.D.). Neben der großen Belastung durch die dauerhafte Dialysebehandlung, die mit mehrmals wöchentlich, stundenlangen, ambulanten Terminen einhergeht, führen die starken Einschränkungen in der Lebensqualität, zum Beispiel durch das Einhalten einer strengen Diät, zu hohem psychischem Stress.

Eine dauerhafte Wiederherstellung der Lebensqualität ist nur durch eine Nierentransplantation möglich. In den USA sind im Jahr 2019 mehr als 20.000 Nieren transplantiert worden. Somit ist die Niere das am häufigsten transplantierte Organ (*Organ Procurement and Transplantation Network* o.D.). Die Erfolgsaussichten nach einer Nierentransplantation ein normales Leben führen zu können steigen durch den medizinischen Fortschritt und der Optimierung des Allokationsprozesses, der die Spenderniere dem Empfänger zuordnet. In Abbildung 1.1 sind die Entwicklung der Anzahl der Nierentransplantationen und die Erfolgsaussichten pro Jahr in den USA dargestellt.

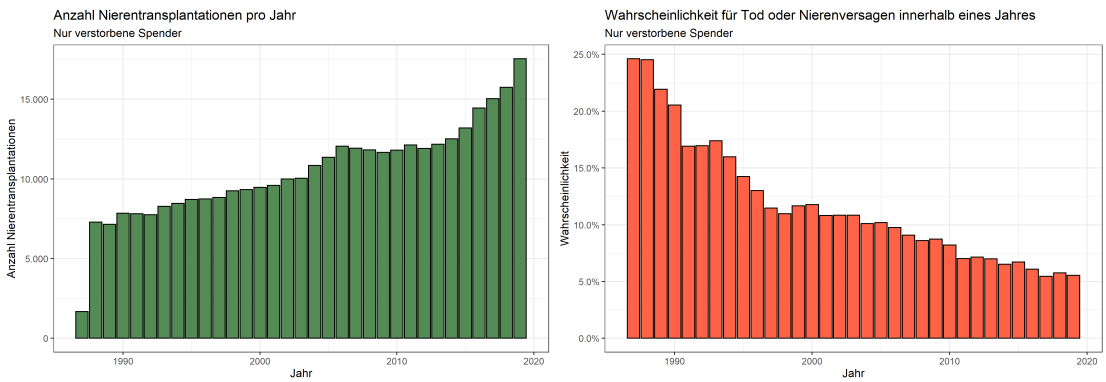


**Abb. 1.1:** Die Entwicklung der Anzahl von Nierentransplantationen pro Jahr (links) und der Wahrscheinlichkeit für Tod oder erneutem Nierenversagen des Empfängers innerhalb eines Jahres (rechts). Das Transplantationsdatum ist zwischen 1987 und 2019 und es wurden nur Transplantationen mit verstorbenen Spendern betrachtet.

Im aktuellen Allokationsprozess spielen der Kidney Donor Profile Index (KDPI) und der Estimated Post Transplant Survival (EPTS) eine zentrale Rolle. Diese kombinieren jeweils mehrere medizinische Faktoren des Spenders bzw. Empfängers miteinander und geben eine Prozentzahl wieder. Hierbei steht ein niedriger KDPI für hochwertige Spendernieren und ein niedriger EPTS für vielversprechende Empfänger. Die Spender und Empfänger werden in mehrere Stufen, entsprechend ihres KDPI und EPTS, eingeteilt. Anschließend werden die Organe auf Basis der verschiedenen Stufen und weiteren Aspekten den Patienten zugewiesen. Der KDPI und EPTS spielen also eine zentrale Rolle für den Erfolg der Organstransplantation, wie in Abbildung 1.2 zu sehen ist (*The New Kidney Allocation System* o.D.).

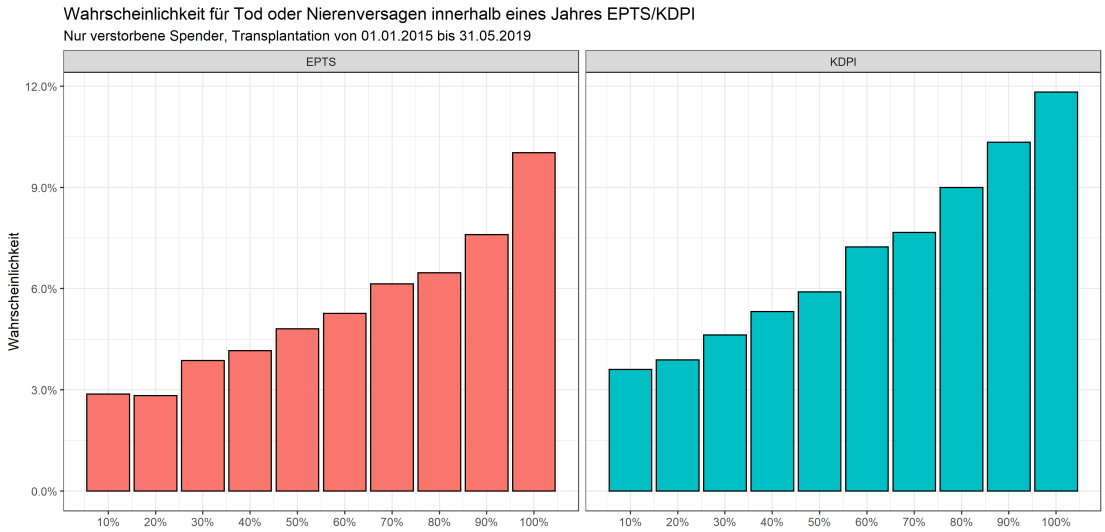


**Abb. 1.2:** Wahrscheinlichkeit für Tod oder erneutem Nierenversagen innerhalb eines Jahres für die Dezile des KDPI (links) bzw. EPTS (rechts). Es wurden nur Transplantationen vom 01.01.2015 bis 31.05.2019 betrachtet.

Machine Learning Modelle, wie beispielsweise neuronale Netze oder Random Forests, bieten zwar eine hohe Vorhersagegenauigkeit, aber das Ergebnis ist anschließend nicht erklärbar. Diese Modelle, auch Black Box Modelle, sind durch den komplexen Einfluss vieler Variablen auf die Vorhersage nicht mehr erklärbar. Bei Entscheidungen, die das Leben eines Menschen gravierend beeinflussen können, sollen Machine Learning Modelle nur unterstützend betrachtet werden. Um den Zweck dieser Modelle voll ausschöpfen zu können, muss der Einfluss der Variablen auf das Resultat für den Betrachter nachvollziehbar sein (Rudin 2019).

In dieser Thesis werden regularisierte additive Regressionsmodelle betrachtet. Diese zeichnen sich durch eine gute Erklärbarkeit aus. Durch zahlreiche Modifikationen der Basisvariante lassen sich die Modelle an unterschiedliche Anwendungsfälle anpassen. In einem Anwendungsfall wird die Überlebenswahrscheinlichkeit nach Nierentrans-

plantation für Empfänger vorhergesagt. Die Daten des Empfängers und Spenders sowie der Transplantation und weitere Informationen werden vom U.S. Department of Health and Human Services erfasst und zur Verfügung gestellt.

Als spezieller Fokus werden die Interaktionen zwischen der Nierenqualität (KDPI) und den Merkmalen des Empfängers untersucht. Die Nierenqualität spielt einen zentralen Aspekt im Allokationsprozess und ist häufig der Grund ein Organ abzulehnen. Es wird untersucht ob der KDPI bei allen Patienten den gleichen Einfluss hat, oder ob mangelnde Nierenqualität in Interaktion mit Merkmalen des Empfängers zu einer anderen Prognose führen.

Bei Analysen zu Nierentransplantionen ist es meistens das Ziel die Wahrscheinlichkeit für eine ereignisfreie Zeit für den Empfänger zu prognostizieren. Ereignisfreie Zeit heißt in diesem Kontext üblicherweise, dass der Patient weder verstorben noch erneutes Nierenversagen erlitten hat. Die Analysen sollen medizinisches Fachpersonal dabei unterstützen gefährdeten Patienten bereits frühzeitig mehr medizinische Betreuung zur Verfügung zu stellen (Kabore *et al.* 2017).

## 1.2 Ziel der Arbeit

Die Interaktionen von Variablen werden in vielen Analysen nicht betrachtet, da sich die Dimensionalität und die damit verbundene Komplexität vervielfacht. In dieser Thesis werden Methoden untersucht, die Interaktionen zu einer ausgewählten Variable (hier: KDPI) unter hoher Dimensionalität aufzeigen können. Dazu werden spezielle, regularisierte additive Modelle trainiert und verglichen, um damit einflussreiche Variablen und Interaktionen zu identifizieren. Das Ziel ist es, auf Basis der Variablenselektion, ein gut erklärbares Regressionsmodell zu trainieren, um ein besseres Verständnis für Einflussfaktoren bei Nierentransplantationen entwickeln zu können. Insbesondere die Interaktion der Nierenqualität (KDPI) mit Merkmalen wie Alter, Geschlecht und Diabeteserkrankung des Empfängers, ist hier von Interesse.

Hierfür müssen zunächst geeignete Verfahren identifiziert und die Unterschiede evaluiert werden. Der Fokus liegt in erster Linie nicht nur auf einer hohen Vorhersagegenauigkeit, sondern auch auf guter Erklärbarkeit und effizienter Identifikation einflussreicher Haupt- und Interaktionseffekte. Um ein besseres Verständnis für die Modelle zu entwickeln, werden diese zusätzlich auf simulierten Daten geprüft. Die Methoden sollen hierbei schwache Interaktionen und Einflüsse einzelner Variablen unter hoher Dimensionalität mit vielen Störfaktoren identifizieren. Hierbei ist es besonders interessant die Einschränkung der hierarchischen Bedingungen zu untersuchen.

Durch den Allokationsprozess entstehen bereits große systematische Zusammenhänge in den Daten, die keine kausale Interpretation von Effektschätzern erlauben. Um mit diesen Störfaktoren zurechtzukommen, müssen die Modelle bzw. Daten entsprechend angepasst werden. Zudem müssen im Zusammenhang mit Interaktionen im Allgemeinen hierarchische Bedingungen beachtet werden. Die selektierten Variablen sollen in einem finalen Regressionsmodell persistiert und dieses mittels einer R-Shiny App verfügbar gemacht werden. Hierbei ist es wichtig, dass die Ergebnisse mittels interaktiver Grafiken und Metriken auch ohne statisches Fachwissen gut verständlich sind.

.

## 1.3 Gliederung

Um ein grundlegendes Verständnis zu entwickeln, werden zunächst einige theoretische Grundlagen in Kapitel 2 erklärt. Dabei geht es insbesondere sowohl um die Grundlagen der Regressionsanalyse mit Interaktionen, als auch um die Probleme mit systematischen Zusammenhängen in den Daten.
Im Rahmen dieser Thesis werden vier spezielle, additive Modelle betrachtet. Die Funktionsweise, Unterschiede und der besondere Fall im Umgang mit Interaktionen werden in Kapitel 3 erläutert.
In Kapitel 4 werden die Nierentransplantationsdaten untersucht. Nach vorheriger Analyse und Vorbereitung der Daten werden die speziellen Regressionsmodelle trainiert. Es wird beschrieben, wie das beste Modell gewählt wurde und dieses in einer R-Shiny App dargestellt wird.
Anschließend werden die Methoden in Kapitel 5 systematisch miteinander verglichen. Hierbei werden sowohl die einzelnen Ergebnisse des Anwendungsfalls evaluiert als auch die Methoden auf simulierten Daten geprüft.
Im abschließenden Fazit in Kapitel 6 werden die Erkenntnisse und Problematik dieser Thesis zusammengefasst und ein Ausblick beschrieben.

# 2 Theoretische Grundlagen

Zu Beginn werden einige theoretische Grundlagen vorgestellt. Diese werden im weiteren Verlauf der Arbeit angewandt. Unter anderem bauen die speziellen Verfahren in Kapitel 3 auf diesen Grundlagen auf.

## 2.1 Die logistische Regression

Die logistische Regression bildet das zentrale Modell dieser Arbeit. Üblicherweise versteht man hierunter die binäre logistische Regression, bei der die abhängige Variable $Y_i$ binäre Ausprägung hat. Die unabhängigen Variablen $x_{ik}$ können dabei ein beliebiges Skalenniveau haben. Kategorielle Variablen werden typischerweise als eine Reihe von Dummy Variablen aufgelöst, die für die jeweilige Ausprägung mit einer null bzw. eins kodiert werden. Eine kategorielle Variable mit einer Vielzahl unterschiedlicher Ausprägungen führt somit zu vielen Dummy Variablen in dem Regressionsmodell.

Mit Hilfe der sogenannten Logitfunktion wird der lineare Prädiktor $\beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k$ für Wahrscheinlichkeiten in ein [0,1]-Intervall skaliert.

$$logit(\theta_i) = ln\left(\frac{\theta_i}{1-\theta_i}\right) = \beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k$$

Wobei $\theta_i := P(Y = 1|X = x_i)$ die Wahrscheinlichkeit zur $i$-ten Beobachtung ist. Durch Umformen nach $\theta_i$ ergibt sich:

$$\theta_i = \frac{\exp(\beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k)}{1 + \exp(\beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k)}$$

Mittels dieses Terms kann das logistische Regressionsmodell definiert werden:

$$P(Y = 1|X = x_i) := \frac{\exp(\beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k)}{1 + \exp(\beta_0 + \sum_{k=1}^{p} x_{ik}\beta_k)}$$

Das Ziel ist es, die Wahrscheinlichkeit, dass $Y_i$ eingetreten ist, unter der Bedingung, dass $X = x_i$ beobachtet wurde, vorherzusagen. Die Schätzung der Parameter $\beta_k$ erfolgt üblicherweise mit einem Maximum-Likelihood-Schätzer (Hastie *et al.* 2009, S. 119).

Zur Interpretation der Koeffizienten wird die Chance (engl. odds) definiert:

$$\text{odds}(x) := \frac{P(Y = 1|X = x)}{P(Y = 0|X = x)},$$

Für stetige Variablen gilt damit:

$$\frac{\text{odds}\left((x_1, \ldots, x_j + 1, \ldots, x_p)\right)}{\text{odds}\left((x_1, \ldots, x_p)\right)} = \exp(\beta_j)$$

Erhöht sich der Wert einer stetigen Variable um 1 und alle anderen Variablen bleiben gleich, so lässt sich das als eine Änderung der Chance um den Faktor $\exp(\beta_j)$ interpretieren. Bei kategoriellen Variablen wird eine Referenzkategorie definiert. Jede Ausprägung der Variable, mit Ausnahme der Referenzkategorie, erhält jeweils einen Koeffizienten. Die Chance des Ereignisses unter der Ausprägung $X_j = A$ im Vergleich zur Referenzkategorie, wenn alle anderen Variablen gleich bleiben, ändert sich um den Faktor $\exp(\beta_{j_A})$. Der Einfluss einer Variable lässt sich in der logistischen Regression also gut nachvollziehen. Zudem kann das Modell zur Klassifikation genutzt werden, indem ein Schwellenwert bestimmt wird. Ist die geschätzte Wahrscheinlichkeit von $Y_i = 1$ größer als diesem Schwellenwert, wird die Beobachtung mit eins klassifiziert, andernfalls null.

Es gibt eine Vielzahl von Metriken um die Performance von logistischen Regressionsmodellen zu untersuchen. Hierbei sind der spezifische Anwendungsfall und die Eigenschaften des Datensatzes wichtig. Bei der Vorhersage von Wahrscheinlichkeiten eignet sich der Brier Score (BS):

$$BS = \frac{1}{n} \sum_{i=1}^{n} (p_i - o_i)^2$$

Dabei ist $p_i$ die vorhergesagte Wahrscheinlichkeit zur $i$-ten Beobachtung. Das eingetretene Ereignis wird von $o_i$ beschrieben und ist null oder eins, je nachdem ob das Ereignis eingetreten ist oder nicht. Zur besseren Interpretierbarkeit lässt sich der BS mittels eines Referenzscores $BS_{ref}$ umwandeln zum Brier Skill Score (BSS):

$$BSS = 1 - \frac{BS_{fit}}{BS_{ref}}$$

Der Referenzscore $BS_{ref}$ kann beispielsweise als Prädiktor gewählt werden, der lediglich die Klassenverteilung der Zielgröße für jede Beobachtung voraussagt. Ein BSS zwischen null und eins ist eine Verbesserung der Vorhersagegenauigkeit gegenüber dem Referenzmodell, wobei eins die perfekte Vorhersage für alle Beobachtungen ist und ein

höherer Wert eine bessere Performance bedeutet. Ein Wert kleiner null bedeutet, dass das Modell schlechter ist als das Referenzmodell (Roulston 2007).

## 2.2 Bias-Variance Tradeoff

Die Vorhersagegenauigkeit mit Daten, die zum Erstellen des Modells benutzt wurden ("Trainingsdaten"), ist im Allgemeinen weniger interessant, da hier das Ergebnis bereits bekannt ist. Bei der Evaluation aller Arten von Modellen ist es daher besonders relevant den Vorhersagefehler auf neuen Daten ("Testdaten") zu untersuchen. Die Testdaten sind unbekannt für das Modell und wurden nicht zum trainieren benutzt. Für ein besseres Verständnis des Zusammenhangs von Test- und Trainingsfehler betrachtet man den erwarteten Testfehler. Hierbei ist $y_i = f(x_i) + \varepsilon$, wobei $\varepsilon$ mit Erwartungswert null und der Varianz $\sigma^2$ als nicht reduzierbarer Fehler bezeichnet wird. Die Funktion $f(x)$ entspricht dem tatsächlichen Modell der Daten und ist üblicherweise unbekannt. Die Funktion $\hat{f}$ kann dabei ein beliebiges Modell sein um $f$ zu schätzen. Die Vorhersage zum Punkt $x_0$ wird beschrieben als $\hat{f}(x_0)$. Es gilt:

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\varepsilon)$$

Dabei definiert $E\left(y_0 - \hat{f}(x_0)\right)^2$ den erwarteten Testfehler. Dieser ist klein, falls sowohl die Varianz als auch der Bias von $\hat{f}(x_0)$ klein sind. Der Fehler $\varepsilon$ ist unabhängig vom Modell und der erwartete Testfehler kann daher nicht niedriger als die Varianz des nicht reduzierbaren Fehlers sein.

Durch die Wahl anderer Trainingsdaten kommt es zu Änderungen im Modellschätzer $\hat{f}$. Diese Schwankungen werden durch die Varianz von $\hat{f}$ beschrieben. Im Idealfall schwankt das Modell bei kleinen Änderungen in den Trainingsdaten nur minimal. Der Bias beschreibt den Fehler komplexe, reale Zusammenhänge in einem einfachen Modell zusammenzubringen. Die Varianz und der Bias stehen also im Gegensatz zueinander. Zum einen soll das Modell komplizierte Zusammenhänge präzise darstellen und zum anderen sich generalisiert auf neue Daten übertragen lassen. Diese Wechselwirkung wird als "Bias-Variance Tradeoff" bezeichnet (James *et al.* 2013, S. 34). Um den Zusammenhang zu verdeutlichen, wird das simulierte Beispiel in Abbildung 2.1 betrachtet.

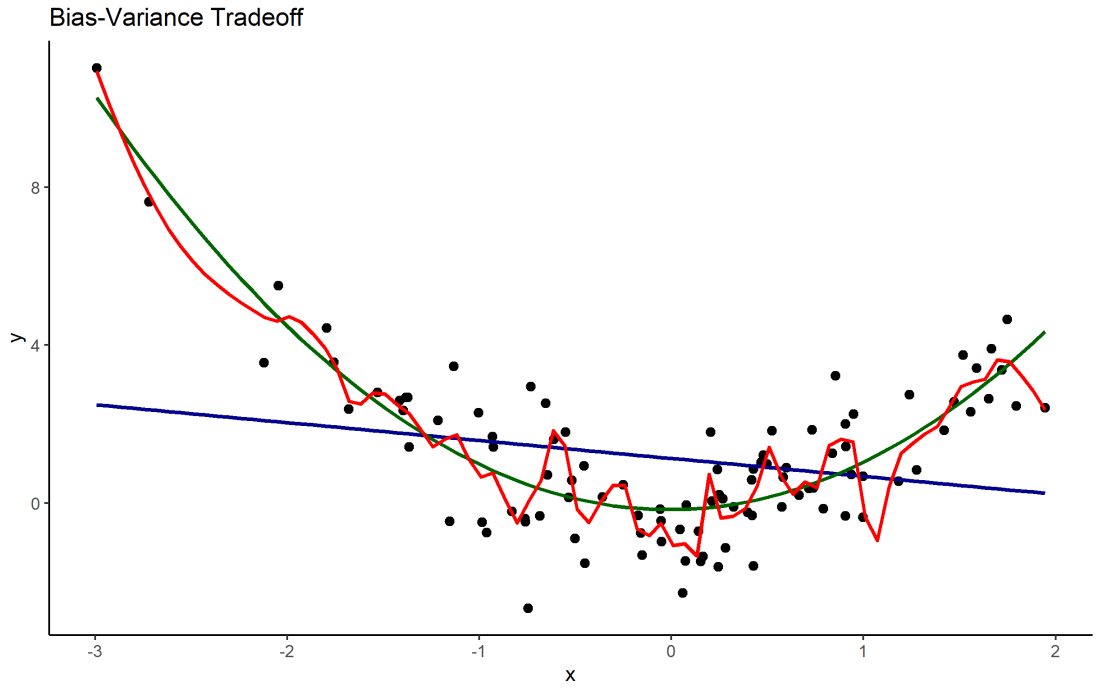


**Abb. 2.1:** Ein simuliertes Beispiel mit $y = x^2 + \varepsilon$. In Blau wurde nur ein linearer Zusammenhang von $x$ auf $y$ modelliert. Die Varianz ist zwar niedrig, aber es gibt eine sehr hohe Abweichung vom tatsächlichen Zusammenhang zwischen $x$ und $y$. In Rot ein zu stark angepasstes Modell. Eine kleine Änderung im Datensatz führt zu einer großen Änderung im Modell, also hohe Varianz. Das passende Modell ist in Grün dargestellt. Dies hat den besten Bias-Variance Tradeoff.

## 2.3 Die Shrinkage Methoden

Die Überanpassung (engl. overfitting) ist ein übliches Problem bei der Regression insbesondere mit vielen Variablen. In Modellen ohne Regularisierung werden alle Variablen mit vollem Effekt aufgenommen, dadurch wird das Modell zu stark an die Daten, mit welchen es trainiert wurde, angepasst. Die Folge ist eine sehr hohe Varianz und damit auch kein optimaler erwarteter Testfehler. Wünschenswert wäre es den Effekt der weniger relevanten Variablen zu verkleinern bzw. Variablen ohne Einfluss gänzlich aus dem Modell zu entfernen. Das Ziel ist, es den erwarteten Testfehler zu reduzieren, indem die Varianz verkleinert wird, ohne den Bias zu stark wachsen zu lassen (Tibshirani 1996).

Die Shrinkage Methoden sind eine Möglichkeit mit Überanpassung umzugehen. Dabei werden die Koeffizienten mittels eines Strafterms verkleinert ("geschrumpft").

### 2.3.1 Die Lasso Regression

Least Absolute Shrinkage and Selection Operator (Lasso) ist eine Shrinkage Methode. Diese ist definiert durch:

$$\beta^{lasso} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$
$$\text{u. d. N.} \sum_{j=1}^{p} |\beta_j| \leq t$$

Die Summe der Koeffizienten $|\beta_j|$ darf den Grenzwert $t$ nicht überschreiten. Um die Nebenbedingung erfüllen zu können, müssen die Koeffizienten betraglich kleiner werden. Um den Fehlerterm weiterhin zu minimieren, werden Variablen, die das Modell nur geringfügig verbessern, stärker geschrumpft. Bei Lasso können Koeffizienten auf null gesetzt und Variablen somit komplett aus dem Modell entfernt werden. Die Nebenbedingung wird auch als Strafterm bezeichnet und hängt vom Parameter $t$ ab. Eine kleinerer Parameter $t$ führt zu einer stärkeren Bestrafung und somit betraglich kleineren (bzw. weniger) Parametern. Die gängige Darstellung der Lasso Schätzung ist in Lagrangeform:

$$\beta^{lasso} = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

Der Parameter $\lambda$ hat einen dem Parameter $t$ entgegengesetzten Einfluss. Ein größerer Wert resultiert in einer stärkeren Bestrafung. Bei den Shrinkage Methoden wird der Parameter $\lambda$ auch als Tuning-Parameter bezeichnet.

### 2.3.2 Die Ridge Regression

Eine weitere Shrinkage Methode ist die Ridge Regression:

$$\beta^{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

Die Ridge Regression unterscheidet sich nur durch den quadratischen Strafterm von Lasso. Der quadratische Strafterm führt dazu, dass Koeffizienten zwar geschrumpft, aber nicht gleich null gesetzt werden können. Diese Eigenschaft wird in der Abbildung 2.2 dargestellt.

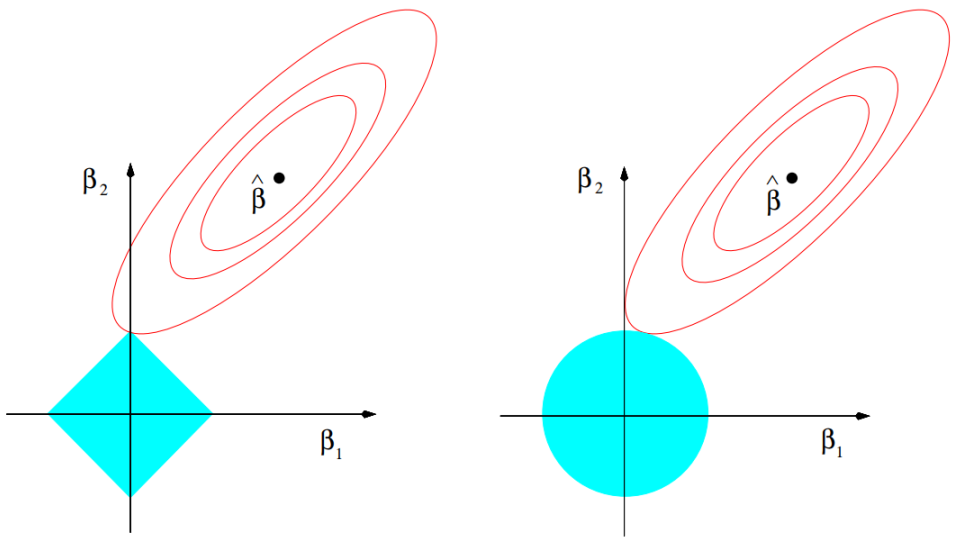


**Abb. 2.2:** Links die Schätzung von Lasso und rechts von Ridge mit zwei Koeffizienten $\beta_1$ und $\beta_2$. Die blauen Flächen zeigen die Nebenbedingungen $\beta_1 + \beta_2 \leq t$ (Lasso) bzw. $\beta_1 + \beta_2 \leq t^2$ (Ridge). Die roten Ellipsen sind die Konturen der Fehlerfunktion. Die Lösung ist die erste Schnittstelle zwischen der blauen Fläche und der Ellipse. Wegen der eckigen Form kann die Lösung von Lasso in einer Ecke liegen und somit wird einer der Parameter auf null gesetzt. Quelle: (Hastie *et al.* 2009, S.71)

Ridge eignet sich gut um die Komplexität von Modellen und die Probleme der Überanpassung zu reduzieren. Für Variablenselektion ist Ridge jedoch ungeeignet, da die Koeffizienten $\beta^{ridge}$ nicht auf null geschrumpft werden können. In der Praxis wird häufig eine Folge $\lambda_1 > \lambda_2 > \ldots \lambda_k$ von Tuning-Parametern betrachtet. Für große Parameter werden die Koeffizienten $\beta_k$ mehr geschrumpft. Dies führt zu einer Verkleinerung der Varianz des Modells, aber einem höheren Bias. Der beste Tuning-Parameter wird im Allgemeinen mit der Kreuzvalidierung festgestellt (Hastie *et al.* 2009, S. 61-71).

## 2.4 Die Kreuzvalidierung

Die Kreuzvalidierung ist die bekannteste Methode auf Überanpassung zu untersuchen und verschiedene Modelle zu vergleichen. Hierbei werden die Beobachtungen im Datensatz zufällig gemischt und in $K$ ungefähr gleich große Teilmengen zerlegt. Anschließend wird das Modell auf $K - 1$ Teilmengen trainiert und der Testfehler auf der verbleibenden Teilmenge bestimmt. Das Ganze wird für alle $K$ Möglichkeiten wiederholt. Zwei Schritte einer Kreuzvalidierung sind in der Abbildung 2.3 dargestellt.

**Abb. 2.3:** Zwei Schritte einer Kreuzvalidierung mit $K = 5$. Im ersten Schritt entsprechen die Teilmengen 1-4 den Trainingsdaten und auf 5 wird das Fehlermaß bestimmt. Im nächsten Schritt entsprechen die Teilmengen 1-3 & 5 den Trainingsdaten und auf 4 wird das Fehlermaß bestimmt.

Das Fehlermaß $L(y, \hat{f}^{-k}(x))$ ist die Abweichung zwischen den vorhergesagten Werten aus dem Modell, das ohne die $k$-te Partition geschätzt wurde, und der $k$-ten Partition des Datensatzes. Aus den Fehlermaßen kann der Kreuzvalidierungsfehler $\mathrm{CV}(\hat{f})$ berechnet werden:

$$\mathrm{CV}(\hat{f}) = \frac{1}{K} \sum_{k=1}^{K} L\left(y_i, \hat{f}^{-k}(x_i)\right)$$

Das Fehlermaß $L(y_i, f(x_i))$ kann dabei beliebig gewählt werden.

Die Wahl von $K$ ist abhängig vom speziellen Anwendungsfall. Ein höheres $K$ führt tendenziell zu einem niedrigerem Bias, aber einer höheren Varianz der Modelle. In der Praxis wird üblicherweise $K = 5$, $K = 10$ oder $K = n$ gewählt. Der Kreuzvalidierungsfehler ist abhängig von der Einteilung der Daten, daher wird die Methode für verschiedene Einteilungen des Datensatz mehrfach durchlaufen. Anschließend kann daraus der durchschnittliche Vorhersagefehler der Iterationen berechnet werden. Der Fall $K = n$ wird auch als Leave-One-Out-Kreuzvalidierung bezeichnet und ist nicht abhängig von der Einteilung des Datensatzes (Hastie *et al.* 2009, S. 241).

## 2.5 Der Bootstrap

Die Bootstrap Methode ist ein Resampling Verfahren und dient der Bestimmung der statistischen Genauigkeit. Hierbei werden aus einem Datensatz $Z = (z_1, \ldots, z_N)$ Stichproben generiert, indem $N$-mal mit zurücklegen zufällig gezogen wird. Für jede dieser

Stichproben wird dann die Kennzahl $S(Z)$ berechnet. Die Kennzahl $S(Z)$ kann dabei eine beliebige Kennzahl sein. Damit lassen sich verschiedene Eigenschaften der Verteilung wie beispielsweise die Varianz:

$$\widehat{Var}[S(Z)] = \frac{1}{B-1}\sum_{b=1}^{B}(S(Z^{*b}) - \bar{S}^{*})^2$$

oder die Konfidenzintervalle von $S(Z)$ schätzen. (Hastie *et al.* 2009, S. 249)

Im Idealfall sollten so viele Stichproben, wie technisch möglich, erstellt werden. Da die Bootstrap Methode einfach zu implementieren und sich universell anwenden lässt, ist sie ein konventionelles Verfahren für die Evaluierung statistischer Kennzahlen. Sowohl in der Inferenzstatistik (z.B. für Hypothesentests), als auch in Machine Learning (z.B. Untersuchung der Stabilität von Modellen), findet die Methode Anwendung.

## 2.6 Interaktionen zwischen Variablen

Nach Lim & Hastie wird von einer Interaktion in der Funktion $f$ von den Variablen $x$ und $y$ gesprochen, wenn sich die Funktion $f(x,y)$ nicht ausdrücken lässt als $g(x)+h(y)$ für zwei Funktionen $g$ und $h$ (Lim & Hastie 2013). In Regressionsmodellen versteht man darunter den multiplikativen Zusammenhang von zwei Variablen $X_i$ und $X_j$ auf die Zielgröße $Y$. Ein einfaches Beispiel, um die Situation zu verdeutlichen:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \theta_{1,2} X_1 X_2 + \varepsilon$$

Die Terme $\beta_1 X_1$ und $\beta_2 X_2$ werden Hauptterme mit den Haupteffekten $\beta_1$ bzw. $\beta_2$ und $\theta_{1,2} X_1 X_2$ als Interaktionsterm mit Interaktionseffekt $\theta_{1,2}$ bezeichnet. Die Hauptterme beeinflussen die Zielvariable hierbei unabhängig voneinander. Der Interaktionsterm von $X_1$ ist abhängig davon, welche Ausprägung $X_2$ hat und umgekehrt. Interaktionseffekte lassen sich mittels Interaktionsplots wie in Abbildung 2.4 veranschaulichen und nachvollziehen.

Im Allgemeinen können bei der Variablenselektion Haupteffekte entfernt werden, aber der zugehörige Interaktionseffekt nicht. Nach Cox führen große Haupteffekte eher zu nennenswerten Interaktionseffekten. Zudem haben Interaktionseffekte für große Haupteffekte mehr praktische Relevanz (Cox 1984) . Daher wird in der Praxis häufig die schwache oder starke Hierarchiebedingung zur Restriktion benutzt. Hierbei können Interaktionen nur dann selektiert werden, wenn auch der entsprechende Haupteffekt einer oder beider Variablen im Modell enthalten ist.

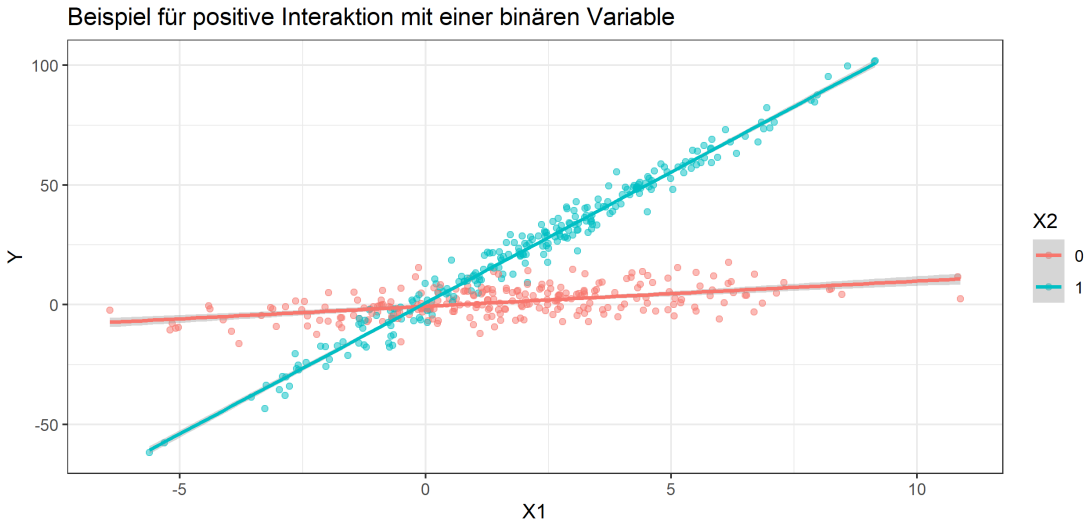


**Abb. 2.4:** Ein simuliertes Beispiel für eine Interaktion. Die Variable $X_1$ ist stetig, und die Variable $X_2$ ist binär. Tritt $X_1$ ohne $X_2$ auf, ist der Effekt auf die Zielgröße $Y$ nur leicht positiv; mit $X_2$ zusammen stark positiv, also eine positive Interaktion. Es wurde mittels der Gleichung $Y = X_1 + X_2 + 10 \cdot (X_1 X_2) + \varepsilon$ simuliert.

Die genaue Definition der Hierarchiebedingungen lautet:

1. *Schwache Hierarchie:* $\theta_{j,k} \neq 0 \longrightarrow \beta_j \neq 0$ oder $\beta_k \neq 0$
2. *Starke Hierarchie:* $\theta_{j,k} \neq 0 \longrightarrow \beta_j \neq 0$ und $\beta_k \neq 0$

Vor allem in medizinischen Studien sind Interaktionen sehr interessant. Es kann beispielsweise das zeitgleiche Auftreten verschiedener Symptome zu einer anderen Prognose führen. Die Anzahl zusätzlicher Parameter durch Interaktionen ist $\binom{p}{k}$, wobei $p$ die Anzahl an Spalten in der Designmatrix und $k$ die Ordnung der betrachteten Interaktionen beschreibt. Mit wachsender Ordnung steigt die Komplexität der Modelle sehr schnell und die Zusammenhänge der Interaktionen sind schwierig zu interpretieren, daher werden üblicherweise nur paarweise Interaktionen ($k = 2$) betrachtet (Bien *et al.* 2013).

## 2.7 Probleme mit Confounding

In medizinischen Studien ist es häufig von Interesse den Effekt einer abhängigen Variable Z (im weiteren Verlauf Behandlung genannt) auf die unabhängige Variable Y (im weiteren Verlauf Resultat genannt) zu untersuchen. Es wird hierbei zwischen behandelten Probanden der Experimentalgruppe und unbehandelten Probanden der Kontrollgruppe unterschieden. Störfaktoren, auch Confounder (C) genannt, haben einen Einfluss sowohl auf das Resultat, als auch auf die Behandlung und verzerren

somit den Effekt. Man spricht von Confounding, wenn sich die Verteilungen der Confounder zwischen der Experimental- und Kontrollgruppe wesentlich voneinander unterscheiden.

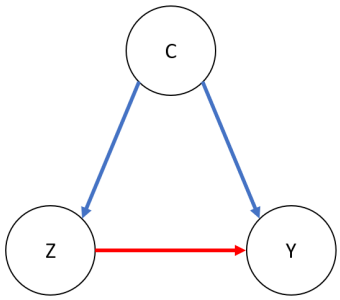


**Abb. 2.5:** Die Confounder (C) haben sowohl auf die Behandlung (Z), als auch auf das Resultat (Y) einen Einfluss. Die Beziehung von Z auf Y wird dabei durch C verzerrt.

Folgendes Beispiel verdeutlicht die Problematik:

In einer Studie soll der Einfluss von Rauchen (Z) auf die Lebenserwartung (Y) ermittelt werden. Einige Eigenschaften (C) der Probanden unterscheiden sich systematisch zwischen Rauchern und Nichtrauchern. Es könnten Raucher gegenüber Nichtrauchern auch eher übergewichtig sein und mehr Alkohol trinken. Diese Faktoren haben ebenfalls einen starken Einfluss auf die Lebenserwartung und verzerren somit den kausalen Effekt von Rauchen auf die Lebenserwartung.

Eine Möglichkeit den kausalen Effekt einer Behandlung auf das Resultat zu untersuchen, ist eine Randomized Controlled Trial (RCT). Dabei werden die Probanden zufällig der Experimental- und Kontrollgruppe zugewiesen. Durch die zufällige Verteilung unterscheiden sich die Eigenschaften der Probanden in der Experimental- und Kontrollgruppe nicht systematisch voneinander. Dieses Vorgehen ist bei vielen Studien aus ethischer Sicht nicht vertretbar, so könnte man im vorherigen Beispiel keine Personen auffordern mit dem Rauchen anzufangen. Zudem sind beobachtende Studie einfacher durchzuführen und es können mühelos signifikant mehr Daten erhoben werden. Der Propensity Score (PS) bietet eine Vielzahl von Möglichkeiten eine beobachtende Studie zu entzerren (Austin 2011).

### 2.7.1 Der Propensity Score

In der Basisvariante des PS wird die Behandlung als ein binäres Ereignis definiert. Die Dosis oder Menge einer Behandlung wird nicht betrachtet. Zunächst sind einige Definitionen nötig:

- $Z_i \in \{0,1\}$: Behandlung von Patient $i$
- $Y_i(0)$ bzw. $Y_i(1)$: Potentielle Resultate des Patienten $i$ mit Behandlungsstatus 0 bzw. 1
- $X_i$: beobachtete Kovariaten, mögliche Confounder

Offensichtlich ist es nur möglich ein Resultat $Y_i(0)$ bzw. $Y_i(1)$ für einen Patienten zu beobachten. Die Behandlung, die der Patient nicht erhalten hat, wird auch als kontrafaktische (engl. counterfactual) Behandlung bezeichnet. Das zugehörige Resultat unter dieser Behandlung wird auch als kontrafaktisches oder potentielles Resultat bezeichnet. In vielen Studien ist es von Interesse den kausalen Effekt der Behandlung zu untersuchen. Hierbei ist es das Ziel den Unterschied zwischen den potentiellen Resultaten $Y_i(0)$ und $Y_i(1)$ zu bestimmen (Höfler 2005).

Der PS ist definiert durch:

$$e_i = P(Z_i = 1|X = x_i).$$

Der PS ist also die Wahrscheinlichkeit, dass Proband $i$ die Behandlung erhält, unter der Bedingung seiner möglichen Confounder. Wenn die Kovariaten eines Probanden stark von den Kovariaten der Probanden in der Experimentalgruppe abweichen, ist der PS niedrig. Der PS kann als ein Ausgleichswert verstanden werden:

$$X \perp\!\!\!\perp Z|E = e \text{ mit } E = P(Z = 1|X)$$

Diese heuristische Formulierung bedeutet, dass für Probanden mit dem selben PS die Verteilung der Kovariaten innerhalb der beiden Gruppen gleich ist. Zudem müssen zwei Bedingungen gelten:

$$(Y(1), Y(0)) \perp\!\!\!\perp Z|X \tag{2.1}$$

$$0 < P(Z = 1|X) < 1 \tag{2.2}$$

Die Bedingung (2.1) sagt aus, dass die potentiellen Resultate unter der Bedingung der möglichen Confounder unabhängig von der Behandlung sein sollen. Anders ausgedrückt sollen alle möglichen Variablen mit systematischen Unterschieden zwischen

Experimental- und Kontrollgruppe in der Menge X enthalten sein. In der Praxis ist diese Bedingung nur eingeschränkt umzusetzen, da selten Daten zu allen möglichen Confoundern erhoben werden. Die Bedingung (2.2) bedeutet, dass die Wahrscheinlichkeit, unter der Bedingung der Kovariaten, die Behandlung zu erhalten größer null und kleiner eins sein soll. Das heißt jeder Proband hat unabhängig seiner Kovariaten die Möglichkeit behandelt bzw. nicht behandelt zu werden (Rosenbaum & Rubin 1983).

Es werden nun einige Methoden zur Anwendung des PS beschrieben:

- **Matching:** Beim Matching werden paarweise Patienten aus der Experimental- und Kontrollgruppe mit einem möglichst ähnlichen PS gezogen. Das Matching ist üblicherweise eins zu eins. Es wird also ein Proband aus der Experimentalgruppe und ein Proband aus der Kontrollgruppe gezogen. Die systematischen Unterschiede zwischen den zwei Gruppen werden damit angeglichen und es kann wie mit einer RCT weitergearbeitet werden, um den Effekt der Behandlung zu untersuchen. Die Grundgesamtheit wird beim Matching allerdings stark verzerrt, da viele Probanden keinem anderen zugeordnet werden können und somit entfallen. Bei der Interpretation der Ergebnisse muss dieser Aspekt beachtet werden.

- **Inverse Gewichtung:** Bei der Inversen Gewichtung wird allen Probanden auf Basis ihres PS und ihrer Gruppenzugehörigkeit ein Gewicht zugeordnet. In der Experimentalgruppe wird dabei mit dem Inversen des PS $1/e_i$ und in der Kontrollgruppe mit $1/(1-e_i)$ gewichtet. Die Probanden mit stark zu ihrer eigenen Gruppe abweichenden Kovariaten werden damit höher gewichtet. Bei vielen Modellen können anschließend diese Gewichte genutzt werden, um die Population anzugleichen.

- **Adjustierung:** Bei der Adjustierung wird ein Modell trainiert mit $Y = \beta_0 + \beta_1 Z + \beta_2 \text{PS} + \varepsilon$. Die Modellwahl ist dabei abhängig von der Zielvariable $y$. Der Effekt der Behandlung lässt sich aus dem Koeffizienten $\beta_1$ ablesen und ist zu verstehen als der Teil der Zielvariable $y$ der sich nicht bereits durch den PS erklären lässt.

Die Wahl der Methode ist abhängig vom speziellen Anwendungsfall. Die Berechnung des PS erfolgt typischerweise über eine logistische Regression (Austin 2011).

### 2.7.2 Generalized Propensity Score

Der Generalized Propensity Score (GPS) ist eine Erweiterung des PS auf stetige Behandlungen. Es wird also nicht nur unterschieden, ob der Patient eine Behandlung erhalten

hat, sondern auch in welcher Dosis bzw. Menge. Zunächst müssen die Definitionen für eine stetige Behandlung angepasst werden:

- $Z_i \in [z_0, z_1]$: Behandlung von Patient $i$ ($[z_0, z_1]$ beliebiges Intervall)
- $Y_i(z), z \in Z$: Potentielles Resultat des Patienten $i$ mit Behandlung $z$

Auch die erste Bedingung 2.1 aus der Basisvariante des PS muss angepasst werden:

$$Y(z) \perp\!\!\!\perp Z|X \quad \forall z \in Z$$

Es wird nicht die gemeinsame Unabhängigkeit für alle Resultate $\{Y(t)\}_{z\in[z_0,z_1]}$, sondern die bedingte Unabhängigkeit für jeden Wert der Behandlung gefordert. Auch hier müssen wieder alle Confounder in den Daten enthalten sein. Anschließend kann der GPS definiert werden. Sei die auf die Kovariaten bedingte Dichte der Behandlung $r(z,x)$ gegeben durch die Gleichung:

$$r(z,x) = f_{Z|X}(z|x) \tag{2.3}$$

Dann ist der GPS definiert als:

$$R = r(Z,X)$$

Auch die stetige Variante hat eine ausgleichende Eigenschaft. Damit gilt:

$$X \perp\!\!\!\perp 1_{Z=z}|r(z,X)$$

Diese heuristische Formulierung bedeutet, dass für den selben GPS die Wahrscheinlichkeit, dass $Z = z$ gilt, unabhängig von den beobachteten Kovariaten ist (Hirano & Imbens 2004).

Robins et al. legt nahe, dass die bedingte Dichte in Gleichung 2.3 normalverteilt ist (Robins *et al.* 2000). Um diese zu bestimmen, wird zunächst die Behandlung auf die Kovariaten regressiert:

$$Z = \beta_0 + \sum_{i=1}^{p} \beta_i X_i + \varepsilon \tag{2.4}$$

Anschließend kann der GPS mittels der Wahrscheinlichkeitsdichtefunktion der Normalverteilung bestimmt werden:

$$\begin{aligned} f(z|X = x) &= f(z|\mu_x, \sigma_x^2) \\ &= \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\left(-\frac{(z-\mu_x)^2}{2\sigma_x^2}\right) \end{aligned}$$

Hierbei entspricht der Parameter $\mu_x$ dem geschätzten Wert $\hat{z}_i$ und der Parameter $\sigma_x^2$ der geschätzten Residuenvarianz aus dem Modell 2.4. Beide Parameter sind abhängig von der jeweiligen Beobachtung $x$. Es können auch andere Modelle als die lineare Regression trainiert werden, um den GPS zu bestimmen. Die Wahl ist abhängig von den Struktur der Daten (Austin 2018).

Einige Methoden der Basisvariante des PS lassen sich nicht auf den GPS übertragen. Eine gängige Methode ist die Adjustierung nach den Confoundern (Dinga *et al.* 2020). Hierzu wird eine weiteres lineares Regressionsmodell trainiert:

$$Z = \beta_0 + \beta_1 GPS + \varepsilon$$

Mit Hilfe der Residuen $\varepsilon_{Z|GPS}$ dieses Modells können die Daten adjustiert werden. Ein übliches Vorgehen ist es dabei Modelle mit $\varepsilon_{Z|GPS}$ und dem GPS oder $Z$ und dem GPS zu trainieren. Im Kontext von Variablenselektion kann es vorkommen, dass ein Teil der Confounder nicht im Modell enthalten ist. Um hierbei den Effekt von $Z$ nicht zu überschätzen, muss $Z$ adjustiert werden. Daher werden mit den Residuen $\varepsilon_{Z|GPS}$ statt $Z$ Modelle trainiert. Es wird demnach nur der Teil der Behandlung betrachtet, der sich nicht bereits (linear) durch den GPS erklären lässt. Die Modelle büßen einerseits Effizienz ein, da die Variable $Z$ nicht mehr enthalten ist, andererseits werden die Ergebnisse weniger durch Confounding kausal beeinflusst .

# 3 Spezielle Verfahren zur Variablenselektion

Bei den speziellen Verfahren handelt es sich um additive Regressionsmodelle mit Lasso Regularisierung. Das Ziel ist es mittels dieser Verfahren Variablen zu selektieren. Diese Verfahren wurden anhand mehrerer Merkmale ausgesucht:

- Anwendbarkeit für die Daten
- Implementierung in R für die logistische Regression
- Umgang mit Interaktionen

Xiaohan Yan & Jacob Bien verschaffen einen guten Überblick über die verschiedenen Methoden in ihrem Artikel *Hierarchical Sparse Modeling: A Choice of Two Group Lasso Formulations* (Yan & Bien 2017). Eine Zusammenfassung der Unterschiede der Verfahren ist am Ende dieses Kapitels zu finden in Tabelle 3.1. In den folgenden Unterkapiteln werden die Verfahren aus Übersichtlichkeit anhand der linearen Regression mit der quadratischen Verlustfunktion erläutert. Die Methodik lässt sich bei allen Verfahren ebenso für die Verlustfunktion der logistischen Regression übertragen.

## 3.1 Group Lasso

Das Group Lasso Verfahren ist eine Erweiterung der Lasso Regression. Hierbei können Gruppen für die unabhängigen Variablen definiert werden. Für die Gruppen gilt, dass entweder alle Koeffizienten innerhalb einer Gruppe gleich null oder alle ungleich null sind. Es können also nicht einzelne Variablen innerhalb einer Gruppe selektiert werden. Dies ist vor allem für kategorielle Variablen, die als Dummy Variablen kodiert wurden, nützlich. Bei der Lasso Regularisierung werden häufig einzelne Ausprägungen von Kategorien selektiert. Das führt zum einen zu fachlichen Problemen. Es ist es oft nicht sinnvoll oder nachvollziehbar warum einzelne Ausprägungen relevant sind und andere nicht. Zum anderen gibt es auch technische Probleme, denn die Lasso Lösung ist stark davon abhängig, wie Dummy Variablen kodiert sind. Die Wahl einer anderen Referenzkategorie kann somit zu anderen Ergebnissen führen (Meier *et al.* 2008).

Für einen Vektor $\boldsymbol{\eta} \in \mathbb{R}^d, d \geq 1$ und eine symmetrische $d \times d$ positive definite Matrix $\boldsymbol{K}$ sei:

$$\|\boldsymbol{\eta}\|_{\boldsymbol{K}} = (\boldsymbol{\eta}^T \boldsymbol{K} \boldsymbol{\eta})^{1/2}.$$

Group Lasso ist dann definiert als die Lösung von:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p}\left\{\left\|Y-\sum_{j=1}^{J}X_j\beta_j\right\|_2^2+\lambda\sum_{j=1}^{J}\|\beta_j\|_{\boldsymbol{K}_j}\right\}$$

Die ursprüngliche Designmatrix $\boldsymbol{X}$ wurde in $J$ Gruppen aufgeteilt. Des Weiteren wird für jede Gruppe $j \in \{1,\ldots,J\}$ eine Kernel-Matrix $\boldsymbol{K_j}$ definiert. Die einfachste Wahl ist die Einheitsmatrix $\boldsymbol{K_j} = \boldsymbol{I_{p_j}}$, wobei $p_j$ die Anzahl der Variablen in Gruppe $j$ ist. Der Strafterm innerhalb einer Gruppe entspricht der $\ell^2$ Norm, daher können einzelne Variablen innerhalb einer Gruppe zwar geschrumpft, aber nicht gleich null gesetzt werden (vgl. Ridge Regression in 2.3.2). Die Gruppen an sich werden nur summiert, daher können die Koeffizienten von allen Variablen einer Gruppe gleichzeitig auf null gesetzt werden. Wenn jede Variable in einer eigenen Gruppe ist und $\boldsymbol{K_j} = \boldsymbol{I_1}$, entspricht Group Lasso der konventionellen Lasso Regression. Der Parameter $\lambda$ ist ebenfalls ein Tuning-Parameter, bei dem ein höherer Wert zur stärkeren Bestrafung der Koeffizienten führt (Yuan & Lin 2006). Group Lasso hat keinen speziellen Umgang bei der Regression mit Interaktionen und erfüllt somit keine Hierarchiebedingungen.

Die Berechnung von Group Lasso ist im Gegensatz zur Basisvariante deutlich schwieriger, da die Lösungspfade nicht stückweise linear sind. In dieser Arbeit wird das beliebte R-Paket `gglasso` von Yang & Zou verwendet (Yang & Zou 2014). Das Paket bietet die Möglichkeit Variablen aus dem Strafterm zu entfernen. Diese können somit nicht aus dem Modell entfernt werden.

## 3.2 GLINTERNET

Learning interactions through hierarchical group-lasso regularization (GLINTERNET) ist ein von Lim & Hastie entwickeltes Verfahren, um paarweise Interaktionen unter Beachtung der starken Hierarchiebedingung zu trainieren. Das Verfahren basiert auf Group Lasso bzw. auf der Erweiterung Overlapping (überlappend) Group Lasso. Bei der überlappenden Variante können Variablen in mehr als einer Gruppe enthalten sein. Man betrachte hierzu ein Beispiel:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \theta_{1,2} X_1 X_2 \tag{3.1}$$

Um die starke Hierarchiebedingung zu erfüllen werden drei Gruppen $G_1 = \{X_1\}$, $G_2 = \{X_2\}$, $G_3 = \{X_1, X_2, X_1X_2\}$ definiert. Bei der Regularisierung mit Group Lasso können nur ganze Gruppen ein- bzw. ausgeschlossen werden, daher kann das Modell zwar die Haupteffekte ohne den Interaktionsterm enthalten, den Interaktionsterm aber

nicht ohne die Haupteffekte. Hierbei werden für jede Gruppe Koeffizienten für jede Variable berechnet. Im Falle, dass alle Gruppen trotz Regularisierung enthalten sind, erhalten die Variablen $X_1$ und $X_2$ damit. Mit dieser Motivation kann GLINTERNET definiert werden.

Seien $F_1$ und $F_2$ zwei kategorielle Variablen mit $L_1$ bzw. $L_2$ verschiedenen Ausprägungen. Die entsprechenden Designmatrizen mit Dummy Variablen sind gegeben durch $\boldsymbol{X_1}$ und $\boldsymbol{X_2}$. Zu lösen ist

$$\begin{aligned}\underset{\mu,\boldsymbol{\alpha},\tilde{\boldsymbol{\alpha}}}{\operatorname{argmin}} \frac{1}{2} &\left\| \boldsymbol{Y} - \mu \cdot \boldsymbol{1} - \boldsymbol{X_1}\boldsymbol{\alpha_1} - \boldsymbol{X_2}\boldsymbol{\alpha_2} - [\boldsymbol{X_1}\ \boldsymbol{X_2}\ \boldsymbol{X_{1:2}}] \begin{bmatrix} \tilde{\boldsymbol{\alpha}}_1 \\ \tilde{\boldsymbol{\alpha}}_2 \\ \boldsymbol{\alpha_{1:2}} \end{bmatrix} \right\|_2^2 \\ &+ \lambda \left( \|\boldsymbol{\alpha_1}\|_2 + \|\boldsymbol{\alpha_2}\|_2 + \sqrt{L_2\|\tilde{\boldsymbol{\alpha}}_1\|_2^2 + L_1\|\tilde{\boldsymbol{\alpha}}_2\|_2^2 + \|\boldsymbol{\alpha_{1:2}}\|_2^2} \right) \end{aligned} \tag{3.2}$$

u. d. N.

$$\sum_{i=1}^{L_1} \alpha_1^i = 0, \sum_{k=1}^{L_2} \alpha_1^j = 0, \sum_{i=1}^{L_1} \tilde{\alpha}_1^i = 0, \sum_{j=1}^{L_2} \tilde{\alpha}_1^j = 0 \tag{3.3}$$

und

$$\sum_{i=1}^{L_1} \alpha_{1:2}^{ij} = 0 \text{ für festes } j, \ \sum_{j=1}^{L_2} \alpha_{1:2}^{ij} = 0 \text{ für festes } i \tag{3.4}$$

Es ist zu sehen, dass im Optimierungsproblem 3.2 für die Dummy Variablen in den Designmatrizen $\boldsymbol{X_1}$ und $\boldsymbol{X_2}$ jeweils zwei Koeffizientenvektoren $\boldsymbol{\alpha_1}$ und $\tilde{\boldsymbol{\alpha}}_1$ bzw. $\boldsymbol{\alpha_2}$ und $\tilde{\boldsymbol{\alpha}}_2$ bestimmt werden. Der Strafterm:

$$\|\boldsymbol{\alpha_1}\|_2 + \|\boldsymbol{\alpha_2}\|_2 + \sqrt{L_2\|\tilde{\boldsymbol{\alpha}}_1\|_2^2 + L_1\|\tilde{\boldsymbol{\alpha}}_2\|_2^2 + \|\boldsymbol{\alpha_{1:2}}\|_2^2}$$

zeigt einige interessante Eigenschaften von GLINTERNET.

Der Term $\sqrt{L_2\|\tilde{\boldsymbol{\alpha}}_1\|_2^2 + L_1\|\tilde{\boldsymbol{\alpha}}_2\|_2^2 + \|\boldsymbol{\alpha_{1:2}}\|_2^2}$ lässt sich als $\ell^2$ Norm der Koeffizientenvektoren $\tilde{\boldsymbol{\alpha}}_1$, $\tilde{\boldsymbol{\alpha}}_2$ und $\boldsymbol{\alpha_{1:2}}$ verstehen. Somit können die Koeffizientenvektoren $\hat{\tilde{\boldsymbol{\alpha}}}_1$, $\hat{\tilde{\boldsymbol{\alpha}}}_2$ und $\hat{\boldsymbol{\alpha}}_{1:2}$ entweder alle gleich null oder alle ungleich null geschätzt werden. Die Konstanten $L_1$ und $L_2$ dienen dazu, die Koeffizienten gleich zu skalieren.

Die Koeffizientenvektoren können dann geschätzt werden mit:

$$\hat{\boldsymbol{\beta}}_1 = \hat{\boldsymbol{\alpha}}_1 + \hat{\tilde{\boldsymbol{\alpha}}}_1$$

$$\hat{\boldsymbol{\beta}}_2 = \hat{\boldsymbol{\alpha}}_2 + \hat{\tilde{\boldsymbol{\alpha}}}_2$$
$$\hat{\boldsymbol{\theta}}_{1:2} = \hat{\boldsymbol{\alpha}}_{1:2}$$

Damit ergibt sich:

$$\hat{\boldsymbol{\theta}}_{1:2} \neq 0 \Rightarrow \hat{\boldsymbol{\beta}}_1 \neq 0 \text{ und } \hat{\boldsymbol{\beta}}_2 \neq 0$$

Die gewünschte starke Hierarchiebedingung ist somit erfüllt.

Falls keine Interaktion zwischen den Variablen $F_1$ und $F_2$ besteht, dementsprechend $\boldsymbol{\alpha}_{1:2}$ gleich null gilt, werden die Koeffizientenvektoren $\tilde{\boldsymbol{\alpha}}_1$ und $\tilde{\boldsymbol{\alpha}}_2$ ebenfalls auf null geschätzt. Das folgt aus der Ungleichung:

$$\sqrt{L_2 a^2 + L_1 b^2} \geq a + b$$

für die Konstanten $L_1$,$L_2 \geq 2$ und Variablen $a$,$b \geq 0$. Die Haupteffekte in $\boldsymbol{\alpha}_1$ und $\boldsymbol{\alpha}_2$ werden weniger bestraft als in $\tilde{\boldsymbol{\alpha}}_1$ und $\tilde{\boldsymbol{\alpha}}_2$. Daher wählt GLINTERNET nur die Gruppen mit $\boldsymbol{\alpha}_1$ und $\boldsymbol{\alpha}_2$, falls kein Interaktionseffekt vorhanden ist. Falls ein Interaktionseffekt $\boldsymbol{\alpha}_{1:2} > \mathbf{0}$ vorhanden ist, so ist die Ableitung des Strafterms in Bezug auf $\boldsymbol{\alpha}_{1:2}$:

$$\frac{\boldsymbol{\alpha}_{1:2}}{\sqrt{L_2 \|\tilde{\boldsymbol{\alpha}}_1\|_2^2 + L_1 \|\tilde{\boldsymbol{\alpha}}_2\|_2^2 + \|\boldsymbol{\alpha}_{1:2}\|_2^2}}$$

Der Interaktionseffekt $\boldsymbol{\alpha}_{1:2}$ mit den Haupteffekten $\tilde{\boldsymbol{\alpha}}_1$ und $\tilde{\boldsymbol{\alpha}}_2$ lässt den Nenner größer werden und somit die gesamte Ableitung kleiner. Daher werden Interaktionen mit Haupteffekten vor “reinen” Interaktionseffekten entdeckt.

Nach Lim & Hastie ist jedoch für das überlappende Group Lasso die Parametrisierung der Nebenbedingungen schwierig, insbesondere bei einer Vielzahl von Variablen. Eine Darstellung als Group Lasso ohne Nebenbedingungen bietet daher Vorteile. Lim & Hastie zeigen, dass die Lösung des Optimierungsproblem mit Nebenbedingungen aus 3.2 - 3.4 äquivalent zur Lösung des Optimierungsproblems ohne Nebenbedingungen ist:

$$\underset{\mu,\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - \mu \cdot 1 - \boldsymbol{X}_1\boldsymbol{\beta}_1 - \boldsymbol{X}_2\boldsymbol{\beta}_2 - \boldsymbol{X}_{1:2}\boldsymbol{\beta}_{1:2}\|_2^2 + \lambda(\|\boldsymbol{\beta}_1\|_2 + \|\boldsymbol{\beta}_2\|_2 + \|\boldsymbol{\beta}_{1:2}\|_2) \quad (3.5)$$

Hierfür müssen die Koeffizientenvektoren $\boldsymbol{\beta}_1$, $\boldsymbol{\beta}_2$ und $\boldsymbol{\beta}_{1:2}$ unter einer bestimmen Parametrisierung definiert werden. Für ein festes $i$ und $j$ wird der Koeffizientenvektoren

$\boldsymbol{\beta}_{1:2}$ zerlegt:

$$\beta_{1:2}^{ij} = \beta_{1:2}^{\cdot\cdot} + (\beta_{1:2}^{i\cdot} - \beta_{1:2}^{\cdot\cdot}) + (\beta_{1:2}^{\cdot j} - \beta_{1:2}^{\cdot\cdot}) + (\beta_{1:2}^{ij} - \beta_{1:2}^{i\cdot} - \beta_{1:2}^{\cdot j} + \beta_{1:2}^{\cdot\cdot})$$

Anschließend wird $\beta_{1:2}^{ij}$ umgeschrieben als:

$$\beta_{1:2}^{ij} = \tilde{\mu} + \tilde{\alpha}_1^i + \tilde{\alpha}_2^j + \tilde{\alpha}_{1:2}^{ij}$$

mit:

$$\begin{aligned}
\tilde{\mu} &= \beta_{1:2}^{\cdot\cdot} \\
\tilde{\alpha_1}^i &= \beta_{1:2}^{i\cdot} - \beta_{1:2}^{\cdot\cdot} \\
\tilde{\alpha_2}^j &= \beta_{1:2}^{\cdot j} - \beta_{1:2}^{\cdot\cdot} \\
\tilde{\alpha}_{1:2}^{ij} &= \beta_{1:2}^{ij} - \beta_{1:2}^{i\cdot} - \beta_{1:2}^{\cdot j} + \beta_{1:2}^{\cdot\cdot}
\end{aligned}$$

Mit $\boldsymbol{\alpha_1} = \boldsymbol{\beta_1}$ und $\boldsymbol{\alpha_2} = \boldsymbol{\beta_2}$ können die Haupt- und Interaktionseffekte bestimmt werden aus:

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_1 &= \hat{\boldsymbol{\beta}}_1 + \hat{\tilde{\boldsymbol{\alpha}}}_1 \\
\hat{\boldsymbol{\theta}}_2 &= \hat{\boldsymbol{\beta}}_2 + \hat{\tilde{\boldsymbol{\alpha}}}_2 \\
\hat{\boldsymbol{\theta}}_{1:2} &= \hat{\boldsymbol{\alpha}}_{1:2}
\end{aligned}$$

Der Beweis der Äquivalenz der Optimierungsprobleme 3.2 - 3.4 und 3.5 ist im Artikel von Lim & Hastie zu finden. Das Ganze lässt sich auch auf den Fall übertragen, falls $F_1$ und/oder $F_2$ stetige Variablen sind. Eine Implementierung von GLINTERNET ist im gleichnamigen R-Paket der Autoren zu finden (Lim & Hastie 2013).

## 3.3 Pliable

Pliable (deutsch: flexibel) von Tibshirani & Friedman bietet eine Variante der Lasso Regularisierung mit modifizierenden Variablen. Hierbei wird zwischen der typischen Designmatrix der Variablen $X$ und der Designmatrix der modifizierenden Variablen $Z$ unterschieden. Die Variablen von $Z$ können dabei sowohl stetig als auch kategoriell sein. Seien die Designmatrizen gegeben durch $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ und $\boldsymbol{Z} \in \mathbb{R}^{n \times k}$. Pliable trainiert das Modell:

$$\hat{y} = \beta_0 1 + \boldsymbol{Z}\boldsymbol{\theta_0} + \sum_{j=1}^{p} \boldsymbol{X_j}(\beta_j 1 + \boldsymbol{Z}\boldsymbol{\theta_j})$$

$$= \beta_0 1 + \boldsymbol{Z\theta_0} + \boldsymbol{X\beta} + \sum_{j=1}^{p} (\boldsymbol{X_j} \circ \boldsymbol{Z})\boldsymbol{\theta_j}$$

wobei $\boldsymbol{X_j} \circ \boldsymbol{Z}$ für die komponentenweise Multiplikation steht. Zudem ist $\boldsymbol{X_j}$ die $j$-te Spalte der Matrix $\boldsymbol{X}$ und jeder Vektor $\boldsymbol{\theta_j}$ hat die Länge $k$. Der Vektor $\boldsymbol{\theta_j}$ beschreibt eine Interaktion zwischen $\boldsymbol{X_j}$ und allen Spalten in der Matrix $\boldsymbol{Z}$. Es soll eine schwache Hierarchiebedingung zwischen $\boldsymbol{X_j}$ und $\boldsymbol{Z}$ bestehen.
Die Zielfunktion für Pliable ist:

$$J(\beta_0, \theta_0, \beta, \Theta) = \frac{1}{2N} \sum (y_i - \hat{y}_i)^2 + (1-\alpha)\lambda \sum_{j=1}^{p} \left( (\|\beta_j, \theta_j\|_2 + \|\theta_j\|_2 \right) + \alpha\lambda \sum_{j,k} |\theta_{j,k}|_1$$

Die $p \times k$ dimensionale Matrix $\boldsymbol{\Theta}$ enthält die Koeffizienten der Interaktionen, wobei die $j$-te Spalte dem Vektor $\boldsymbol{\theta_j}$ entspricht.

Man betrachte zunächst die Norm $\|\beta_j, \boldsymbol{\theta_j}\|_2$ im ersten Teil des Strafterms. Durch die $\ell^2$ Norm kann der Vektor $\boldsymbol{\theta_j}$ nur ungleich null sein, falls der Koeffizient $\beta_j$ ebenfalls ungleich null ist und umgekehrt. Durch die zusätzliche Summe mit der Norm $\|\theta_j\|_2$ kann der Koeffizient $\beta_j$ auch ungleich null sein obwohl der Vektor $\boldsymbol{\theta_j}$ gleich null ist. Zusammen sorgt der erste Teil des Strafterms damit für die Einhaltung der schwachen Hierarchiebedingung.

Man betrachte nun die Summe $\sum_{j,k} |\theta_{j,k}|_1$. Hier werden die einzelne Elemente von den Vektoren $\boldsymbol{\theta_j}$ in der $\ell^1$ Norm summiert. Dadurch können auch einzelne Interaktionen mit der modifizierenden Variablen geschrumpft bzw. gleich null gesetzt werden. Der Parameter $\lambda$ entspricht dem üblichen Tuning-Parameter in Lasso. Außerdem gibt es noch einen weiteren Tuning-Parameter $\alpha \in [0,1]$. Ein großer Parameter $\alpha$ führt zu einer starken Bestrafung der Interaktionsterme, der Fokus liegt dann auf den Haupteffekten und besonders starken Interaktionen. Im Falle, dass $\alpha$ gegen eins läuft (aber nicht genau eins ist), werden die Interaktionseffekt sehr stark bestraft und somit aus dem Modell entfernt. Pliable entspricht dann der klassischen Lasso Regression ohne Interaktionen.

Weiterhin ist zu beachten, dass der Vektor $\boldsymbol{\theta_0}$ und die Koeffizienten $\beta_0$ nicht bestraft werden. Der Term $\boldsymbol{\theta_0 Z}$ entspricht dem Haupteffekt der modifizierenden Variablen. Tibshirani & Friedman sprechen von optionalen Komponenten die nicht zwangsweise im Modell sein müssen. Enthält das Modell ebenfalls den Vektor $\boldsymbol{\theta_0}$ ist es sogar stark hierarchisch. Pliable ist im gleichnamigen R-Paket implementiert (Tibshirani & Friedman 2018).

## 3.4 GAMSEL

Mit der Methode Generalized Additive Model Selection (GAMSEL) können generalisierte additive Modelle mit hoher Dimensionalität trainiert und regularisiert werden. Das Verfahren bietet als einziges der hier vorgestellten Verfahren die Möglichkeit nicht lineare Effekte zu untersuchen. Chouldechova & Hastie motivieren die Idee mit Smoothing Splines.

Zur Definition des Smoothing Spline wird zunächst der Fall mit einem einzigen Prädiktor $x$ und einer Resultatvariable $y$ betrachtet. Die Funktion $\hat{f}_\lambda$ definiert dann den Smoothing Spline Schätzer für einen Parameter $\lambda > 0$:

$$\hat{f}_\lambda = \underset{f \in \mathbb{C}^2}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int f''(t)^2 dt \tag{3.6}$$

Der Strafterm kontrolliert wie stark die Funktionen $f \in C^2$ varieren darf. Durch einen großen Parameter $\lambda \in (0,\infty)$ nähert sich die Funktion $f$ einer linearen Funktionen an. Das Optimierungsproblem sieht nicht lösbar aus, da die Menge $C^2$ überabzählbar ist. Das Problem lässt sich jedoch auf eine einfache Form reduzieren. Die Lösung von (3.6) ist ein natürlicher kubischer Spline mit Knoten an jedem einzigartigen Punkt von dem Vektor $x$. Damit lässt sich die Funktion $f$ schreiben als:

$$f(x) = \sum_{j=1}^{n} N_j(x)\theta_j, \tag{3.7}$$

wobei $N_j(x)$ die $n$-dimensionalen Basisfunktionen der natürlichen kubischen Splines sind. Das Optimierungsproblem in Gleichung (3.6) reduziert sich damit auf:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{\lambda}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^n}{\operatorname{argmin}} (y - \boldsymbol{N}\boldsymbol{\theta})^T (y - \boldsymbol{N}\boldsymbol{\theta}) + \lambda \boldsymbol{\theta}^{\boldsymbol{T}} \boldsymbol{\Omega}_{\boldsymbol{N}} \boldsymbol{\theta} \tag{3.8}$$

$$= \underset{\boldsymbol{\theta} \in \mathbb{R}^n}{\operatorname{argmin}} \|y - \boldsymbol{N}\boldsymbol{\theta}\|_2^2 + \lambda \boldsymbol{\theta}^{\boldsymbol{T}} \boldsymbol{\Omega}_{\boldsymbol{N}} \boldsymbol{\theta} \tag{3.9}$$

$$= \underset{\boldsymbol{\theta} \in \mathbb{R}^n}{\operatorname{argmin}} \|y - \boldsymbol{U}\boldsymbol{\theta}\|_2^2 + \lambda \boldsymbol{\theta}^{\boldsymbol{T}} \boldsymbol{D}\boldsymbol{\theta}, \tag{3.10}$$

wobei $\{\boldsymbol{N}\}_{ij}$ den Auswertungen der Basisfunktionen $N_j(x_i)$ entspricht. Die Strafmatrix $\{\boldsymbol{\Omega}_{\boldsymbol{N}}\}_{jk}$ enthält die Einträge $\int N_j''(t) N_k''(t) dt$. Im letzten Schritt (3.9) wurde mittels der Demmler-Reinsch Basis $u_k$ umgeformt (Demmler & Reinsch 1975).

Die Spalten $\boldsymbol{u_k}$ sind aus der Matrix $\boldsymbol{U}$ und die Matrix $\boldsymbol{D}$ ist diagonal mit Einträgen $d_k$. Die Smoothing Spline Lösung $\hat{f}_\lambda$ ist dann:

$$\hat{f}_\lambda = \boldsymbol{U}(\boldsymbol{I} + \lambda \boldsymbol{D})^{-1} \boldsymbol{U}^T \boldsymbol{y} \tag{3.11}$$

$$= \sum_{j=1}^{n} \boldsymbol{u_j} \frac{\langle \boldsymbol{u_j}, \boldsymbol{y} \rangle}{1 + \lambda d_j}. \tag{3.12}$$

Die Matrix $\boldsymbol{S_\lambda} = \boldsymbol{U}(\boldsymbol{I} + \lambda \boldsymbol{D})^{-1}\boldsymbol{U}^T$ wird auch als Glättungsmatrix bezeichnet. Da sich durch einen großen Parameter $\lambda$ der Smoothing Spline einer linearen Regression annähert, reduziert sich dadurch der Freiheitsgrad. Der von $\lambda$ abhängige Freiheitsgrad wird auch als effektiver Freiheitsgrad bezeichnet. Der effektive Freiheitsgrad kann mittels der Glättungsmatrix bestimmt werden aus der Gleichung $df_\lambda = \mathrm{Spur}(\boldsymbol{S_\lambda})$

Das Smoothing Spline Optimierungsproblem wird nun erweitert für $p$ unabhängige Variablen:

$$\underset{f_j \in \mathbb{C}^2}{\operatorname{argmin}} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{p} f_j(x_{ij}) \right)^2 + \lambda \sum_{j=1}^{p} \int f_j''(t)^2 dt$$

Mit Hilfe der Umformungen (3.7)-(3.12) lässt sich das Problem umformulieren zu:

$$\underset{\alpha_0 \in \mathbb{R}, \{\boldsymbol{\beta_j} \in \mathbb{R}^{n-1}\}_1^p}{\operatorname{argmin}} \left\| y - \alpha_0 - \sum_{j=1}^{p} \boldsymbol{U_j} \boldsymbol{\beta_j} \right\|_2^2 + \lambda \sum_{j=1}^{p} \boldsymbol{\beta_j}^T \boldsymbol{D_j} \boldsymbol{\beta_j} \tag{3.13}$$

wobei $\boldsymbol{U_j}$ die Demmler-Reinsch Basis zu $x_j$ ist.

Mit Hilfe dieser Vorüberlegungen kann GAMSEL vorgestellt werden.
Seien hierzu die Funktionen $f_j$ definiert als:

$$f_j(x) = \alpha_j x_j + u_j(x_j)^T \beta_j$$

wobei $\boldsymbol{u_j}$ ein Vektor von $m_j$ Basisfunktionen ist. Der Koeffizient $\alpha_j$ entspricht dem linearen Teil und die Koeffizienten $\boldsymbol{\beta_j} \in \mathbb{R}^{m_j}$ dem nicht linearen Teil der Variable $X_j$. Hierbei ist es wichtig zu beachten, dass die Koeffizienten $\boldsymbol{\beta_j}$ ebenfalls eine lineare Komponenten beinhalten, der lineare Teil von der Variable $X_j$ überlappt sich damit in $\alpha_j$ und $\boldsymbol{\beta_j}$. GAMSEL bietet die Möglichkeit einen Freiheitsgrad vor dem Training zu definieren, um die Schwankung der Funktionen zu kontrollieren. Sei $\boldsymbol{U_j} \in \mathbb{R}^{n \times m_j}$ die Matrix der Auswertungen dieser Funktionen.

Die Zielfunktion von GAMSEL ist:

$$\begin{aligned} \underset{\alpha_0, \{\boldsymbol{\alpha_j}\}, \{\beta_j\}}{\operatorname{argmin}} \frac{1}{2} & \left\| \boldsymbol{y} - \alpha_0 - \sum_{j=1}^{p} \alpha_j x_j - \sum_{j=1}^{p} \boldsymbol{U_j} \boldsymbol{\beta_j} \right\|_2^2 \\ & + \lambda \sum_{j=1}^{p} \left( \gamma |\alpha_j| + (1-\gamma) \|\boldsymbol{\beta_j}\|_{D_j^*} \right) + \frac{1}{2} \sum_{j=1}^{p} \psi_j \boldsymbol{\beta_j}^T \boldsymbol{D_j} \boldsymbol{\beta_j}. \end{aligned}$$

wobei $\|\boldsymbol{\beta_j}\|_{D_j^*} = \sqrt{\boldsymbol{\beta_j}^T \boldsymbol{D_j^*} \boldsymbol{\beta_j}}$ gilt.

Die Matrizen $\boldsymbol{D_j}$ und $\boldsymbol{D_j^*}$ sind angepasste Varianten der Diagonalmatrix aus dem Optimierungsproblem 3.13 für jede Variable $X_j$. Das erste Element der Matrix $\boldsymbol{D_j}$ ist null und alle nachfolgenden Elemente sind aufsteigend und positiv. Die Matrix ist so skaliert, dass das zweite Element gleich eins ist. In der Matrix $\boldsymbol{D_j^*}$ wird das erste Element mit einer eins ersetzt, um damit auch eine lineare Komponente in $\boldsymbol{\beta_j}$ darzustellen.

Die Norm $\|\boldsymbol{\beta_j}\|_{D_j^*}$ im ersten Teil des Stafterms sorgt dafür, dass entweder alle Komponenten des Vektors $\boldsymbol{\beta_j}$ gleich null oder alle ungleich null sind, ähnlich wie bei Group Lasso. Durch die Summe der Koeffizienten $|\alpha_j|$ im Strafterm wird der lineare Anteil von $x_j$ regularisiert. Der Parameter $\gamma \in (0,1)$ ist ein weiterer Tuning-Parameter. Ein hoher Wert führt zu einer starken Bestrafung des linearen Teils und bevorzugt einen nicht linearen Effekt. Falls eine Variable $X_j$ nur einen linearen Effekt enthält, wird der Koeffizient $\alpha_j$ gegenüber den Koeffizienten $\boldsymbol{\beta_j}$ bevorzugt für einen moderaten Wert von $\gamma$, da andernfalls nicht lineare Effekte ebenfalls in das Modell gezwungen werden.

Der zweite Teil des Strafterms $\sum_{j=1}^{p} \psi_j \boldsymbol{\beta_j}^T \boldsymbol{D_j} \boldsymbol{\beta_j}$ wird dabei von den Autoren als end-of-path-penalty bezeichnet. Dieser sorgt dafür, dass jede Variable den vorher definierten Freiheitsgraden entspricht. GAMSEL bietet keinen speziellen Umgang mit Interaktionen und erfüllt somit keine Hierarchiebedingung (Chouldechova & Hastie 2015).

Abschließend eine kurze Zusammenfassung über die Unterschiede der Methoden:

| Verfahren | Hierarchie | Umgang mit kategoriellen Variablen | Nicht-Lineare Effekte |
|---|---|---|---|
| Group Lasso | Keine | Ja | Nein |
| GLINTERNET | Stark | Ja | Nein |
| Pliable | Schwach (Stark, falls $\boldsymbol{\theta_0 Z}$ im Modell enthalten) | Nein | Nein |
| GAMSEL | Keine | Nein | Ja |

**Tab. 3.1:** Die wichtigsten Unterschieden der Methoden.

# 4 Anwendung auf Nierentransplantationsdaten

## 4.1 Vorgehen

Die in Kapitel 3 vorgestellten Methoden wurden anhand eines Anwendungsfalls auf Nierentransplantationsdaten untersucht. Das Vorgehen für diesen Anwendungsfall orientierte sich am Cross Industry Standard Process (CRISP) Modell. CRISP bietet eine universelle Vorlage für Data Mining und Analyseprojekte und ist unabhängig von der genutzten Technologie und dem Einsatzgebiet. Es soll somit den Prozess branchenübergreifend vereinheitlichen. Es wird zwischen sechs Phasen, die teilweise abhängig voneinander sind, unterschieden. Die genaue Struktur muss für den speziellen Anwendungsfall bestimmt werden. Da sich aus den Ergebnissen neue Fragen ergeben können, erfolgen die Phasen in der Praxis häufig iterativ und werden mehrmals durchlaufen. Die Zusammenhänge der Phasen sind in Abbildung 4.1 dargestellt.

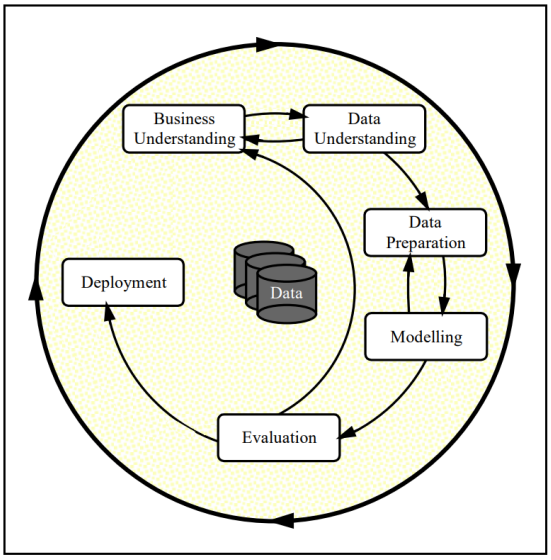


**Abb. 4.1:** Die Phasen von CRISP sind teilweise abhängig voneinander. Der große äußere Kreislauf stellt die iterative Struktur des Prozesses dar, Quelle: (Wirth 2000).

Im folgenden Absatz werden die einzelnen Phasen und ihre Beziehungen zueinander kurz beschrieben:

- **Phase I; Business Unterstanding:** In der ersten Phase soll ein Verständnis für das Ziel des Anwendungsfalls entwickelt werden. Hierbei wird fachliches Wissen zur Branche erworben um den Kontext der Anwendung einordnen zu können. Mit diesem Wissen wird ein vorläufiger Projektplan erstellt.

- **Phase II; Data Understanding:** In dieser Phase wird der Datensatz untersucht. Zum einen geht es dabei um die Herkunft der Daten und zum anderen werden

Erkenntnisse über die Datenqualität und erste Einblicke über interessante Merkmale der Daten gewonnen. In Abbildung 4.1 ist eine Wechselwirkung zwischen den ersten beiden Phasen zu sehen, da sich aus den gewonnenen Informationen über die Daten auch ein besseres Verständnis für den Anwendungsfall entwickelt und umgekehrt.

- **Phase III; Data Preperation:** Im nächsten Schritt werden die Rohdaten aufbereitet. Durch die vorherigen Phasen wird dabei eine passende Untermenge an Beobachtungen und Variablen gewählt. Außerdem werden auch etwaige Probleme mit der Datenqualität behoben.
- **Phase IV; Modelling:** In der vierten Phase werden die Modelle auf den aufbereiteten Daten trainiert. Auch hier besteht eine Wechselwirkung mit der vorherigen Phase. Viele Modelle benötigen spezielle Datenformate oder es ergeben sich weitere Ideen neue Daten zu entwickeln, dadurch müssen die Daten erneut vorbereitet werden.
- **Phase V; Evaluation:** Bevor die Modelle und Ergebnisse einem Anwender zur Verfügung gestellt werden, werden diese evaluiert. Dabei wird festgestellt ob die in Phase I definierten Ziele des Anwendungsfalls erreicht wurden. Durch die Evaluation können sich neue Erkenntnisse ergeben und der Prozess wird erneut durchlaufen.
- **Phase VI; Deployment:** Im letzten Schritt werden die Erkenntnisse für die Anwender verfügbar gemacht. Das Deployment reicht von einem einfachen Bericht bis hin zu komplexen Applikationen. Die neuen Erkenntnisse können beim Anwender zu neuen Fragen führen und der Prozess wird erneut durchlaufen.

(Wirth 2000)
In den folgenden Unterkapiteln wird CRISP auf den Anwendungsfall eingesetzt. Dabei wird für jede Phase das Ziel und Vorgehen erläutert.

## 4.2 Business Understanding

Im ersten Schritt wurde die Phase Business Understanding auf den Anwendungsfall übertragen. Dazu wurde ein grundlegendes Verständnis für die Nierentransplantation und den zugrunde liegenden Allokationsprozess entwickelt. Zudem wurde der Anwendungsfall beschrieben und die Ziele der Analyse festgehalten.

### 4.2.1 Der Allokationsprozess

Der aktuelle Allokationsprozess, New Kidney Allocation System (KAS), wurde im Juni 2013 vom Organ Procurement and Transplantation Network (OPTN) genehmigt und wird seit dem 4. Dezember 2014 klinisch in den USA eingesetzt. Die Fairness und ein großer Nutzen sind dabei die zentralen Aspekte des Prozesses. Zum einen ist es das Ziel, dass Empfänger langfristig ohne weitere Behandlungen auskommen. Zum anderen hat jeder Patient eine faire Chance auf eine Nierentransplantation.

Der KDPI und EPTS spielen bei der Allokation eine wichtige Rolle. Der KDPI setzt sich aus zehn Faktoren des Spenders zusammen: Alter, Größe, Gewicht, ethnische Herkunft, Vorgeschichte von Bluthochdruck und Diabetes, Todesursache, Serumkreatininlevel, Hepatitis C-Status und Spende nach Kreislauftod. Der EPTS hingegen setzt sich aus vier Faktoren des Empfängers zusammen: Alter, Diabetes, vorheriger Transplantation und Dialysebehandlung in Jahren. Sowohl der KDPI als auch der EPTS geben jeweils eine Prozentzahl wieder, wobei niedrigere Werte für ein höherwertiges Organ bzw. einen vielversprechenderen Empfänger stehen.

Im ersten Schritt von KAS wird der KDPI in mehrere Stufen eingeteilt. Anschließend werden Empfänger entsprechend ihres EPTS diesen Stufen zugewiesen. Innerhalb jeder Stufe des KDPI läuft dann ein Priorisierungsprozess ab. Dieser ist abhängig von:

- **Histokompatibilität:** Humane Leukozytenantigene (HLA) sind zentral für das Immunsystem. Antikörper werden durch Antigene vom Immunsystem als Reaktion auf einen Fremdstoff, wie Viren oder ein Organtransplantat, im Körper gebildet. In mehreren Stufen wird unterschieden wie passend die HLA des Empfängers und Spenders zueinander sind. Eine höhere Abweichung erhöht die Gefahr einer Abstoßung des neuen Organs (Schönemann 2016). Die Patienten, deren HLA kompatibel zum Spender sind, werden priorisiert.

- **Blutgruppe:** Durch präventive Maßnahmen, wie die Plasmapherese, lassen sich Organe heutzutage auch mit inkompatiblen Blutgruppen transplantieren. Um das Risiko einer Infektion bereits vor der Transplantation durch diese Therapien zu vermeiden, sollten die Blutgruppen von Spender und Empfänger im Idealfall übereinstimmen (Zschiedrich *et al.* 2015). Die Patienten mit der seltenen Blutgruppe B hatten besonders lange Wartezeiten im vorherigen Allokationsprozess und werden daher im KAS bevorzugt.

- **Geographische Distanz:** Durch weite Transportwege von der Entnahme- zur Transplantationsklinik muss das entnommene Organ lange auf Eis gelagert

werden. Diese Zeitspanne, die auch als kalte Ischämiezeit bezeichnet wird, kann die Niere schädigen und den Empfänger somit gefährden (Sengers *et al.* 2019). Die Empfänger werden daher zunächst lokal zur Entnahmeklinik gesucht.

- **Tage auf der Warteliste:** Ab dem Beginn einer Dialyse oder falls die glomeruläre Filtrationsrate (GFR) unter 20 mL/min fällt, werden die Tage auf der Warteliste gezählt. Bei der Beurteilung der Nierenfunktion spielt die GFR eine wichtige Rolle. Die Patienten, die lange auf der Warteliste stehen, werden priorisiert.
- **Vorherige Lebendspende:** Die Empfänger, die zuvor ein Organ oder Teile eines Organs gespendet haben, werden bevorzugt.
- **Kinder:** Empfänger unter 18 Jahren werden im Prozess stark bevorzugt.

Für jedes dieser Merkmale werden Punkte vergeben, beispielsweise erhalten Empfänger pro Jahr auf der Warteliste einen Punkt. Der Empfänger mit den meisten Punkten erhält anschließend das Organ. Falls kein passender Spender innerhalb der ersten KDPI Stufe gefunden wurde, wird das Organ anderen potentiellen Empfängern verfügbar gemacht. Wird auch in anderen KDPI Stufen kein passender Empfänger gefunden, wird die Niere entsorgt und nicht zur Transplantation genutzt (Chopra & Sureshkumar 2015). Obwohl die Nachfrage deutlich höher als das Angebot ist, werden mehr als 15% der Spendernieren entsorgt (Mohan *et al.* 2018). Der KAS ist ein dynamischer Prozess und wird fortlaufend weiterentwickelt und optimiert (*Policies* o.D.). Eine schematische Zusammenfassung des Allokationsprozesses ist in Abbildung 4.2 zu sehen.

### 4.2.2 Der Anwendungsfall

Das Ziel war es, durch historische Daten über Nierentransplantationen in den USA ein besseres Verständnis für Risikofaktoren des Empfängers und Interaktionen mit der Nierenqualität (KDPI) in Bezug auf die Überlebenswahrscheinlichkeit innerhalb eines Jahres zu entwickeln. Überleben hieß in diesem Kontext, dass der Patient weder verstorben noch erneutes Nierenversagen erlitten hat. Es wurden nur verstorbene Spender betrachtet, da bei der Lebendspende ein anderer Prozess vorausgeht. Die potentiellen Anwender waren medizinisches Personal, die für den Allokationsprozess zuständig sind und Ressourcen für Patienten zur Verfügung stellen. Hierzu wurde mit den in Kapitel 3 beschriebenen regularisierten Verfahren zur Selektion von Haupt- und Interaktionstermen eine Modellwahl getroffen, um mit diesen Variablen anschließend ein einfaches Regressionsmodell zu trainieren. Die Erkenntnisse wurden durch eine R-Shiny App verfügbar gemacht. Die Einflüsse sowohl der Haupt- als auch der

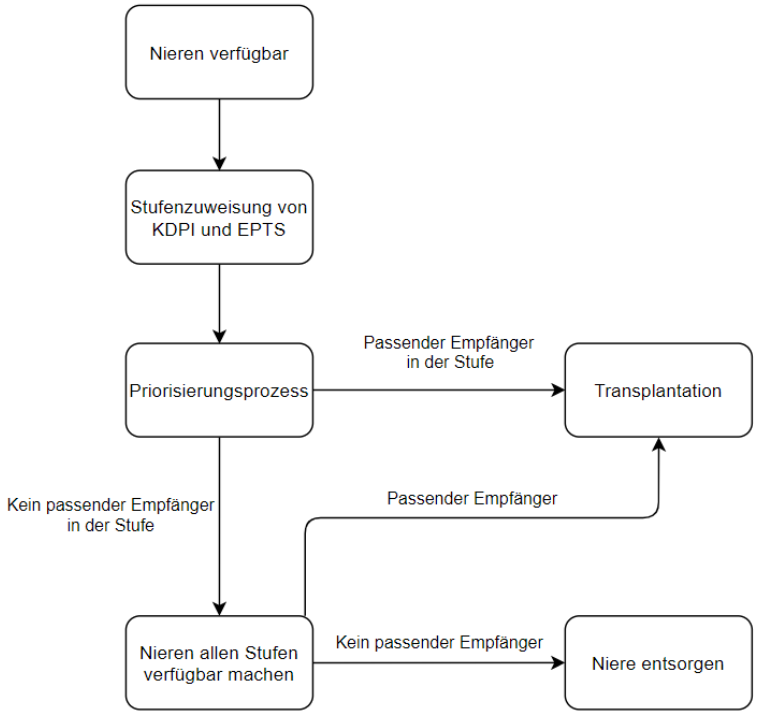


**Abb. 4.2:** Schematische Darstellung des KAS.

Interaktionseffekte mit dem KDPI waren dabei auch ohne statisches Fachwissen verständlich.

## 4.3 Data Understanding

In dieser Phase wurde die Herkunft und Struktur der Daten untersucht. Zudem wurden relevante Daten identifiziert und es wurde ein erster Überblick über die Datenqualität und potentielle Probleme mit Confounding erstellt.

### 4.3.1 Herkunft und Struktur der Daten

Die Transplantationsdaten wurden vom U.S. Department of Health and Human Services über OPTN und das United Network for Organ Sharing (UNOS) bereitgestellt. Der Datensatz enthält nicht nur Informationen zu den Nieren, sondern auch zu anderen Transplantationen, wie beispielsweise Lunge oder Herz. Die Pankreastransplantation ist üblicherweise kein lebensnotwendiger Eingriff, daher wird diese meistens nur in Kombination mit einer Nierentransplantation durchgeführt (Oberkofler & Rougemont 2015). Die Informationen zu beiden Transplantationen liegen daher in den selben Dateien.

Die Daten sind in tabellarischer Form und stehen sowohl als Delimited Text File, als auch als SAS Datensatz zur Verfügung. Hierbei bot sich das SAS Format an, da in diesem die Datenformate der einzelnen Variablen enthalten sind. Zusätzlich wurden noch Dokumentationen mitgeliefert, die zum einen die Relationen der verschiedenen Tabellen erläutern und zum anderen die Variablen kurz beschreiben und kodierte Werte übersetzen. Mit Hilfe der Dokumentation und ersten Einblicken in die Daten wurde sich ein relationales Datenschema abgeleitet. Dieses ist in Abbildung 4.3 dargestellt.

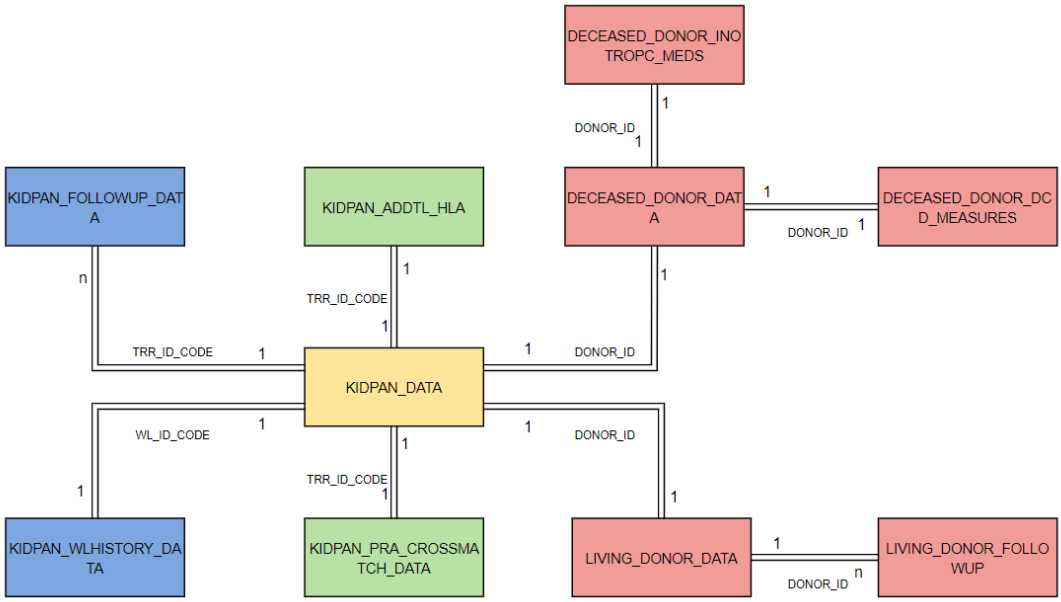


**Abb. 4.3:** Das relationale Datenschema. Im Schema wurden verschiedene Tabellen durch Farben gruppiert. In Gelb ist die zentrale Tabelle `KIDPAN_DATA` mit der sich die restlichen Tabellen verknüpfen lassen. Die roten Tabellen beziehen sich auf den Spender und die blauen auf den Empfänger. Die Tabellen in Grün enthalten Informationen zur Histokompatibilität.

### 4.3.2 Identifikation relevanter Tabellen

Nicht alle Tabellen der Nierentransplantationsdaten enthalten Informationen, die sich für diesen Anwendungsfall eigneten. Die Tabellen wurden daher zunächst auf ihre Relevanz geprüft. Die Farben entsprechen denen aus dem Schema in Abbildung 4.3 und dienen lediglich zur Orientierung.

- **Zentrale Tabelle (Gelb):** Die Tabelle `KIDPAN_DATA` bildet das Zentrum der Nieren- und Pankreastransplantationsdaten. Darin enthalten sind Informationen zum Spender und Empfänger vor der Transplantation und aktuelle Einträge zur letzten Nachuntersuchung. Die Daten der Transplantationen bzw. Wartelisteneinträge

sind seit dem 1. Oktober 1987 verfügbar. Zusätzlich enthalten sind noch Details zu den Rahmenbedingung der Transplantation, wie beispielsweise die geografische Distanz zwischen Entnahme- und Transplantationsklinik. Durch mehrere Schlüssel können weitere Tabellen mit `KIDPAN_DATA` verknüpft werden. Die Tabelle umfasst über eine Millionen Zeilen und 480 Spalten. Die Dimensionalität der Tabelle musste deutlich reduziert werden, bevor Modelle damit lokal trainiert werden konnten.

- **Spender (Rot):** Die roten Tabellen beziehen sich auf den Spender und waren im Rahmen dieses Anwendungsfalls nicht relevant. Es wurden nur der KDPI vom Spender und die Interaktionen mit diesem untersucht. Der KDPI ist bereits in der Tabelle `KIDPAN_DATA` enthalten.

- **Histokompatibilität (Grün):** In den Tabellen `KIDPAN_ADDTL_HLA` und `KIDPAN_PRA_CROSSMATCH_DATA` sind detaillierte Informationen zur Histokompatibilität von Spender und Empfänger enthalten. Viele der Spalten sind ebenfalls in der Tabelle `KIDPAN_DATA` enthalten. Die restlichen Variablen enthalten spezifische Werte zu bestimmten Antigenen und deren Positionen. Hierbei gibt es große Probleme mit der Datenqualität und es gibt nur wenige vollständige Beobachtungen. Die Tabellen in Grün eigneten sich daher nicht für diesen Anwendungsfall und wurden nicht betrachtet.

- **Empfänger (Blau):** Die Tabelle `KIDPAN_WLHISTORY_DATA` enthält Informationen zum Empfänger auf der Warteliste. Die Spalten sind dabei entweder bereits in `KIDPAN_DATA` enthalten oder beschreiben verschiedene Akzeptanzkriterien für Spenderorgane. Darin wird beispielsweise festgehalten wie alt oder schwer ein Spender maximal sein darf, damit das Organ zur Auswahl steht. Die Akzeptanzkriterien waren für diese Auswertung nicht interessant und wurden daher nicht betrachtet. In der Tabelle `KIDPAN_FOLLOWUP_DATA` sind Nachuntersuchungen beim Empfänger nach Transplantation festgehalten. Eine Nachuntersuchung wird erstmalig nach sechs Monaten und anschließend jährlich durchgeführt. Diese Tabelle hätte sich eignen können um den Überlebensstatus des Empfängers festzustellen.

Nach Rücksprache mit dem UNOS Team enthält die Tabelle `KIDPAN_DATA` auch den aktuellsten Status zum Überleben und erneutem Nierenversagen des Empfängers. Damit ist auch die Tabelle `KIDPAN_FOLLOWUP_DATA` zur Bestimmung des Überlebensstatus entfallen und es wurde nur die zentrale Tabelle `KIDPAN_DATA` im Rahmen dieses Anwendungsfall betrachtet. Durch die hohe Dimensionalität, sowohl durch viele Beobachtungen als auch viele Variablen, waren noch einige Einschränkungen vor dem Training der Modelle nötig.

### 4.3.3 Erste Einblicke in die Daten

Um sich einen ersten Überblick über die Datenqualität der Tabelle `KIDPAN_DATA` zu verschaffen, wurden die fehlenden Werte untersucht. Dabei wurde ein Wert als fehlend betrachtet, wenn er "N/A" ist oder einen leeren String enthält. Zusätzlich können Spalten kodierte, unbekannte Werte enthalten. Beispielsweise bedeutet der Wert "998" in der Spalte DIAB, dass der Diabetes Status unbekannt ist. Aus Abbildung 4.4 lässt sich ablesen, dass bei über der Hälfte der Spalten mehr als 70% der Beobachtungen fehlende Werte haben. Fehlende Werte bedeuten jedoch nicht zwangsweise keine Information. Beispielsweise könnte ein fehlender Wert bei der Variable "Dialyse in Jahren" lediglich bedeuten, dass der Patient keine Dialyse erhalten hat. In der nächsten Phase wurde der Datensatz eingeschränkt um mit fehlenden Werten umzugehen.

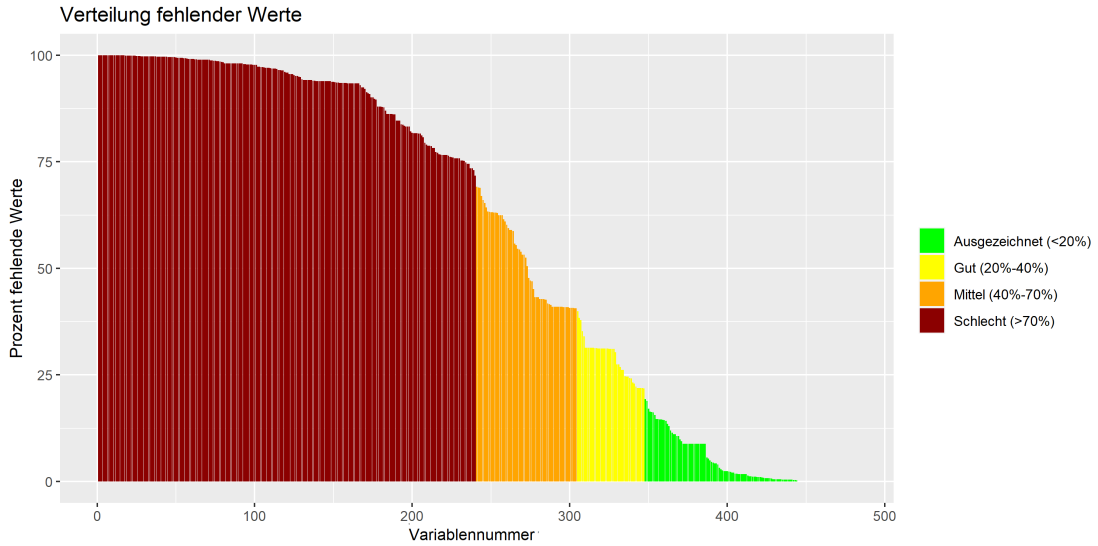


**Abb. 4.4:** Die Verteilung der fehlenden Werten pro Spalte. Auf der y-Achse ist der prozentuale Anteil an fehlenden Werten pro Variable und auf der x-Achse ist die Variablennummer abgebildet. Die Variablennummer wurde absteigend nach der Anzahl fehlender Werte sortiert. Ein Wert wird als fehlend betrachtet, wenn er "N/A" ist oder einen leeren String enthält. Bei mehr als der Hälfte der Variablen fehlen bei mehr als 70% der Einträge die Werte und lediglich 151 Variablen haben eine ausgezeichnete Datenqualität.

Aufgrund des Allokationsprozesses wurde ein Confounding der Variablen des EPTS auf den KDPI vermutet. Der angenommene Zusammenhang ist in Abbildung 4.5 dargestellt. Um einen ersten Eindruck für die möglichen Probleme mit Confounding zu erhalten, wurde der Zusammenhang der KDPI mit dem Alter des Empfängers untersucht. In Abbildung 4.6 ist die Verteilung des Alters unterschieden nach den Stufen des KDPI im KAS dargestellt. Das Alter ist in der Berechnung des EPTS enthalten und beeinflusst somit den KDPI durch den Allokationsprozess. Dies ist auch in den Unterschieden

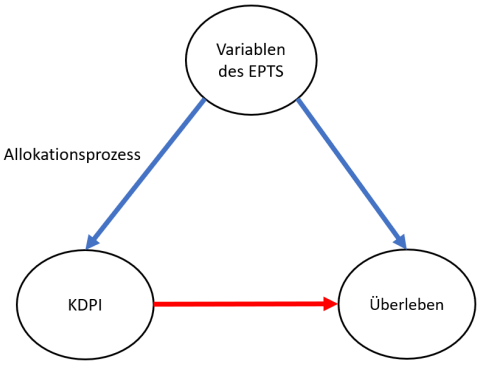


**Abb. 4.5:** Vermutete Probleme mit Confounding. Die Variablen des EPTS haben durch den Allokationsprozess einen Einfluss auf den KDPI. Diese Variablen beeinflussen auch die Überlebenswahrscheinlichkeit des Patienten. Die Beziehung von KDPI auf das Überleben wird dabei durch die Variablen des EPTS verzerrt.

in der Verteilung für die verschiedenen Stufen des KDPI zu sehen. Das Alter beeinflusst ebenfalls die Überlebenswahrscheinlichkeit des Empfängers und führt somit zu Confounding. Es war zu erwarten, dass alle Variablen, die im EPTS enthalten sind, Confounder sind. Um mit dem Confounding umzugehen, musste in der nächste Phase entsprechend adjustiert werden.

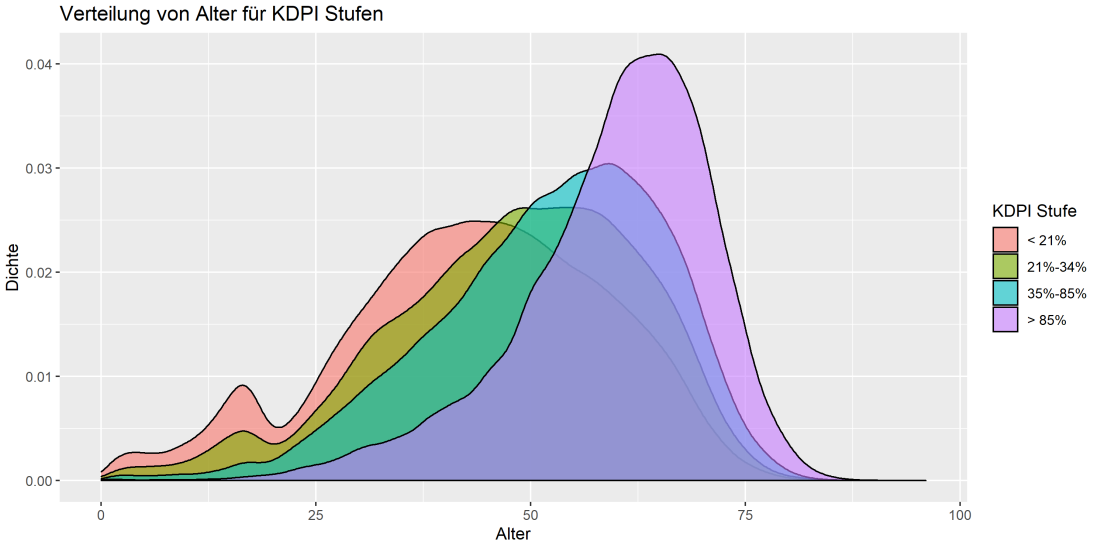


**Abb. 4.6:** Die Verteilung mittels Kerndichteschätzung des Alters unterschieden nach Stufen des KDPI des Allokationsprozesses. Empfänger, die eine Niere aus einer höheren KDPI Stufe erhalten, sind im Schnitt älter. Es liegen Probleme mit Confounding vor.

## 4.4 Data Preperation

In dieser Phase wurde der Datensatz für das Training der Modelle vorbereitet. Dabei wurde sowohl das Patientenkollektiv eingeschränkt als auch Variablen vorausgewählt.

Zudem wurde der Umgang mit Confounding festgelegt und der KDPI entsprechend adjustiert.

### 4.4.1 Einschränkung des Patientenkollektivs

Im ersten Schritt der Datenvorbereitung wurde das Patientenkollektiv eingeschränkt. Die Zielmenge wurde zugeschnitten um sich inhaltlich für den Anwendungsfall zu eignen. Die Einschränkungen sind im folgenden Abschnitt mit einer kurzen Erklärung beschrieben:

- **Transplantationen:** In `KIDPAN_DATA` sind nicht nur Transplantationen enthalten sondern auch potentielle Empfänger, die lediglich auf der Warteliste stehen und bisher noch keine Niere erhalten haben. Die Wartelisteneinträge sind für diesen Anwendungsfall ungeeignet und wurden aus den Daten entfernt.
- **Verstorbene Spender:** Der Prozess für Lebendspende unterscheidet sich gänzlich von dem Allokationsprozess mit verstorbenen Spendern. Die Transplantationen mit Lebendspende wurden daher aus den Daten entfernt.
- **Niere:** Es sind nicht nur Nieren- sondern auch Pankreas- und Kombinationstransplantationen im Datensatz enthalten. Um eine einheitliche Grundmenge zu erhalten, wurden nur reine Nierentransplantationen betrachtet.
- **Datum:** Es wurden Transplantationen ab 01.01.2015 betrachtet, da KAS seit Ende 2014 klinisch eingesetzt wird. Zudem sind Informationen zu Nachbehandlungen bis zum 4. September 2020 in den Daten enthalten. Um den Überlebensstatus innerhalb eines Jahres zu bestimmen wurde das Transplantationsdatum daher zusätzlich bis maximal zum 31. Mai 2019 eingeschränkt. Die Nachuntersuchungen nach einem Jahr kann sich etwas verzögern, daher wurde der Überlebensstatus mit etwas Kulanz innerhalb von 400 Tagen bestimmt.
- **Alter:** Patienten unter 18 Jahren werden stark bevorzugt im Prozess, daher wurde die Grundgesamtheit auf Empfänger ab 18 Jahren eingeschränkt.

Nach den Einschränkungen des Patientenkollektivs wurde der Datensatz von über eine Millionen auf 58.900 Beobachtungen reduziert.

### 4.4.2 Variablenvorauswahl

Viele Variablen sind für den Anwendungsfall und die Modelle ungeeignet und konnten daher bereits vor der Modellierung entfernt werden. Variablen wurden anhand von fünf Kriterien entfernt:

- **Spender:** Es wurden keine Variablen des Spenders, außer dem KDPI und die Interaktionen mit diesem, in diesem Anwendungsfall betrachtet.
- **Pankreas:** Bei einigen Variablen werden nur für Pankreas- und Kombinationstransplantationen Werte erhoben. Diese Analyse beschränkte sich auf Nierentransplantationen.
- **Nicht medizinische Werte:** Nicht medizinische Werte, wie beispielsweise Informationen über das Transplantationszentrum oder die Zahlungsart des Patienten, waren für diese Analyse nicht interessant.
- **Fehlende Werte:** Die Modelle benötigen beim Training einen vollständigen Datensatz. Die Beobachtungen, die fehlende Werte enthalten, wurden daher dabei entfernt. Die Spalten mussten daher möglichst vollständig sein, da ansonsten zu viele Beobachtungen entfallen. Es wurden nur Variablen, die zu mindestens 80% einen Wert haben, selektiert.
- **Viele Ausprägungen:** Kategorielle Variablen mit vielen Ausprägungen, wie beispielsweise Freitextfelder, können zu instabilen Modellen führen. Die kategoriellen Variablen mit mehr als 100 Ausprägungen wurde daher aussortiert.

Zudem werden einige Variablen mehrfach zu verschiedenen Zeitpunkten bestimmt. Beispielsweise wird der BMI bei der Aufnahme auf die Warteliste und vor der Transplantation berechnet. In diesem Fall wurde der Wert gewählt, der zeitlich am nächsten vor der Transplantation ist. Insgesamt wurden 28 Variablen, von denen 16 kategoriell und 12 stetig sind, vorausgewählt.

### 4.4.3 Umgang mit Confounding

Wie bereits in Abbildung 4.6 zu sehen war, entstehen durch den Allokationsprozess systematische Zusammenhänge in den Daten. Im Variablenselektionsprozess können Confounder Effekte dazu führen, dass relevante Einflussfaktoren wie z.B. Alter nicht mehr oder erst spät selektiert werden, wenn KDPI bereits im Modell enthalten ist. Um diesem Phänomen entgegenzuwirken, wurde im Variablenselektionsprozess nicht der

KDPI selbst, sondern nur der Teil des KDPI modelliert, der nicht bereits durch die Confounder erklärt war. Dazu wurde zunächst der GPS bestimmt. Der Allokationsprozess hängt maßgeblich von den Zuweisungen der KDPI Stufen an die entsprechenden EPTS Stufen ab, daher wurde der GPS aus den Variablen des EPTS bestimmt. Die Berechnung des EPTS setzt sich zusammen aus einem linearen Modell mit den Variablen Alter, Diabetes, vorherigen Transplantationen, Dialysebehandlung in Jahren und Interaktionstermen mit Diabetes. Die genaue Formel für die Berechnung des EPTS ist in Abbildung 4.7 zu sehen.

$$\begin{aligned}\text{Raw EPTS} &= 0.047 * max(\text{Age} - 25, 0) - 0.015 * \text{Diabetes} * max(\text{Age} - 25, 0)\\ &+ 0.398 * \text{Prior Solid Organ Transplant} - 0.237 * \text{Diabetes} * \text{Prior Organ Transplant}\\ &+ 0.315 * log(\text{Years on Dialysis} + 1) - 0.099 * \text{Diabetes} * log(\text{Years on Dialysis} + 1)\\ &+ 0.130 * (\text{Years on Dialysis} = 0) - 0.348 * \text{Diabetes} * (\text{Years on Dialysis} = 0) + 1.262 * \text{Diabetes}\end{aligned}$$

**Abb. 4.7:** Die genaue Berechnung des EPTS. Neben den vier Variablen Alter, Diabetes, vorherigen Transplantationen und Dialysebehandlung in Jahren, sind noch Interaktionsterme mit Diabetes enthalten, Quelle: (*A Guide to Calculating and Interpreting the Estimated Post-Transplant Survival* 2020).

Aus dieser Vorüberlegung ergab sich ein Verfahren um den GPS zu bestimmen. Dazu wurde zunächst ein lineares Modell trainiert, bei dem der KDPI auf die Variablen des EPTS regressiert wurde. Anschließend wurde mittels der Wahrscheinlichkeitsdichtefunktion der Normalverteilung der GPS bestimmt. Die genaue Berechnung des GPS ist in Kapitel 2.7.2 zu finden.

Um die ausgleichende Eigenschaft des GPS zu überprüfen (d.h. Alter und KDPI sollten bedingt auf den GPS unabhängig sein), wurde die gewichtete Verteilung mittels Kerndichteschätzung geschätzt (Abbildung 4.8). Durch die Gewichtung mit dem GPS konnten die Verteilungen des Alters innerhalb der vier Stufen des KDPI aneinander angeglichen und somit die systematischen Zusammenhänge reduziert werden.

Mit Hilfe des GPS wurde ein weiteres lineares Modell trainiert. Hierbei wurde der KDPI auf den GPS regressiert. Für die Variablenselektion wurden die Modelle mit den Residuen dieses Modells anstelle des KDPI trainiert. Es wurde somit nur der Teil des KDPI betrachtet, der sich nicht bereits linear durch den GPS erklären lässt. Um Prognosen zu erstellen, wurde im finalen Modell mit dem tatsächlichen KDPI trainiert.

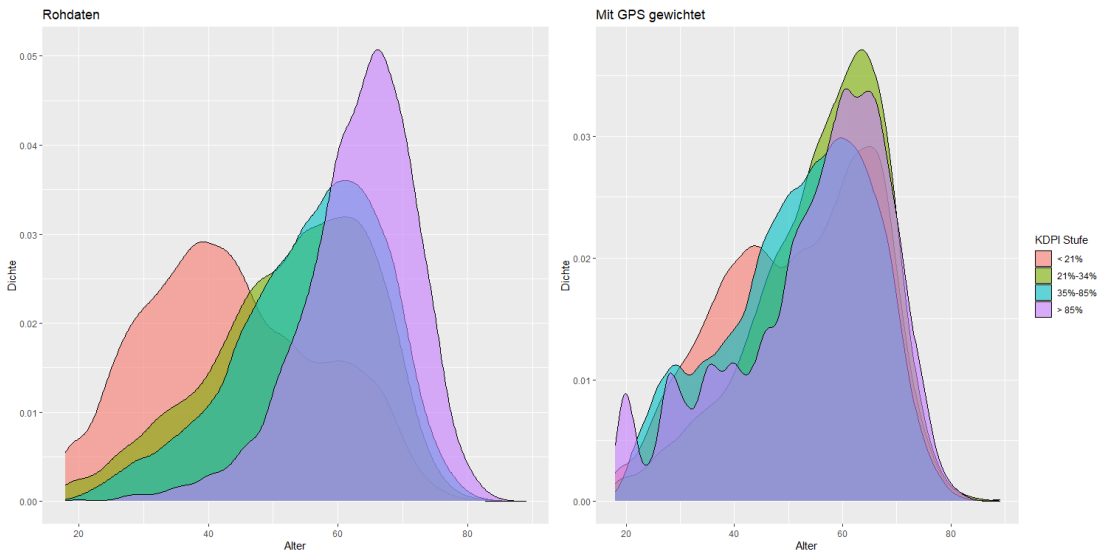


**Abb. 4.8:** Verteilung mittels Kerndichteschätzung von Alter für KDPI Stufen gewichtet mit GPS. Links die Verteilung des Alters der Rohdaten (wie in Abbildung 4.6) , rechts mit Gewichtung auf Basis des GPS.

### 4.4.4 Ergebnis der Datenvorbereitung

Im letzten Schritt vor der Modellierung wurden Beobachtungen mit fehlenden Variablen entfernt, da die Modelle nur mit vollständigen Einträgen trainiert werden konnten. Zusätzlich wurde bei der Variable DIAG_KI, welche die spezifische Diagnose der Nierenerkrankung enthält, Ausprägungen mit wenigen Beobachtungen zusammengefasst. Im finalen Datensatz verblieben 54.566 Beobachtungen. Die Designmatrix, die die kategoriellen Variablen als Dummy Variablen aufgelöst hat, enthielt mit den Interaktionen des KDPI 161 Variablen. Eine Tabelle mit allen selektierten Variablen, dem Variablentyp und einer kurzen Beschreibung ist zu finden in Tabelle 4.1.

| Variable | Typ (Anzahl Kategorien) | Beschreibung |
|---|---|---|
| ABO | Kategoriell (8) | Blutgruppe |
| ABO_MAT | Kategoriell (3) | Abweichung der Blutgruppen |
| AGE | Stetig (-) | Alter |
| AMIS | Kategoriell (3) | Abweichung des A-Antigens |
| BMI_CALC | Stetig (-) | BMI |
| BMIS | Kategoriell (3) | Abweichung des B-Antigens |
| CMV_STATUS | Kategoriell (2) | Cytomegalovirus Erkrankung |
| COLD_ISCH_KI | Stetig (-) | Kalte Ischämiezeit |
| DAYSWAIT | Stetig (-) | Tage auf der Warteliste |
| DIAB | Kategoriell (5) | Diabeteserkrankung |
| DIAB_DUR | Stetig (-) | Diabetesdauer in Jahren |
| DIAG_KI | Kategoriell (34) | Diagnose der Nierenerkrankung |
| DIAL_DUR | Stetig (-) | Dialysedauer in Jahren |
| DRMIS | Kategoriell (3) | Abweichung des DR-Antigens |
| EBV_SEROSTATUS | Kategoriell (3) | Epstein-Barr-Virus Status |
| EGFR_CKDEPI_TRR_PRETX | Stetig (-) | Geschätzter GFR |
| END_CPRA_DETAIL | Stetig (-) | Berechneter PRA |
| ETHCAT | Kategoriell (7) | Ethnische Zugehörigkeit |
| GENDER | Kategoriell (2) | Geschlecht |
| HBV_CORE | Kategoriell (3) | Hepatitis B Antikörper |
| KDPI | Stetig (-) | Residuen des KDPI |
| MALIG | Kategoriell (2) | Vorherige Malignität |
| NUM_PREV_NON_KI_TX | Stetig (-) | Anzahl vorheriger nicht Nierentransplantationen |
| NPKID | Stetig (-) | Anzahl vorheriger Nierentransplantationen |
| ORG_REC_ON | Kategoriell (2) | Lieferart des Organs |
| PERIP_VASC | Kategoriell (2) | Periphere arterielle Verschlusskrankheit |
| SERUM_ALBUM | Stetig (-) | Serumalbumin |
| TX_PROCEDUR_TY_KI | Kategoriell (4) | Transplantationstyp |

**Tab. 4.1:** Die vorausgewählten Variablen. Die Einschränkungen basieren sowohl auf fachlicher Relevanz für den Anwendungsfall als auch auf technischen Einschränkungen durch fehlende Werte bzw. zuvielen Ausprägungen bei kategoriellen Variablen.

## 4.5 Modelling

In dieser Phase wird sowohl das Vorgehen bei der Modellierung als auch die Bestimmung des besten Modells erläutert.

### 4.5.1 Vorgehen der Modellierung

Vor dem Training der Modelle mit den vier Verfahren wurde ein einheitliches Vorgehen mit vergleichbaren Parametern festgelegt. Ein übliches Vorgehen bei der Lasso Regularisierung ist es mittels Kreuzvalidierung den besten Tuning-Parameter zu bestimmen. Die R-Pakete Pliable und GGLasso warfen jedoch nicht lösbare Fehler bei der Ausführung der Kreuzvalidierungsfunktion für den Datensatz aus.

Ein alternativer Ansatz, der sich für alle Methoden einheitlich anwenden lies, war es mit Hilfe von Bootstrapping die Modelle zu untersuchen. Dabei war es das Ziel eine Rangliste für relevante Variablen und Interaktionen zu erstellen. Alle Verfahren wurden auf 50 Bootstrap Stichproben trainiert. Auf allen Stichproben wurde eine Folge von 50 Tuning-Parametern $\lambda_k$ ($\lambda_1 > \ldots > \lambda_{50}$) festgelegt. Mit diesem Vorgehen konnten zum einen relevante Haupt- und Interaktionsterme aus dem Schnitt selektiert und zum anderen die Methoden im nächsten Kapitel systematisch miteinander verglichen werden.

Für einige der Verfahren wurde der Datensatz noch an ein spezielles Datenformat angepasst und Parameter zum Training wurden voreingestellt:

- **GGLasso:** Bei kategoriellen Variablen und Interaktionen mit diesen, wurden die Dummy Variablen in der Designmatrix entsprechend gruppiert. Die Variable KDPI wurde mittels eines Strafterms in das Modell gezwungen und damit nicht regularisiert. Hiermit ist die schwache Hierarchiebedingung erfüllt, da KDPI immer im Modell enthalten war.

- **GLINTERNET:** Vor dem Training musste festgelegt werden, welche Variablen kategoriell sind und wieviele Ausprägungen diese haben. Zusätzlich war es nötig, Kategorien in eine aufeinanderfolgende Zahlenfolge umzukodieren. In der Standardeinstellung untersucht GLINTERNET alle paarweisen Interaktionen. Für den hier beschriebenen Anwendungsfall wurde diese Einstellung angepasst, um nur Interaktionen mit dem KDPI zu modellieren.

- **GAMSEL:** Für die binären Dummy Variablen und für stetige Variablen mit wenigen Ausprägungen wurde festgelegt, dass hier nur ein linearer Zusammenhang

betrachtet wurde. Für alle weiteren stetigen Variablen wurde die Standardeinstellung mit zehn Freiheitsgraden festgelegt. Der Tuning-Parameter $\gamma$ wurde auf 0,4 belassen, um einen linearen Zusammenhang bei der Regularisierung leicht zu bevorzugen.

- **Pliable:** Die optionalen Komponenten der modifizierenden Variable (hier: KDPI) und des Intercepts wurden dem Modell hinzugefügt, da diese zentral für alle Modelle sind. Der KDPI wurde somit nicht regularisiert und war immer im Modell enthalten. Damit war die starke Hierarchiebedingung bei Pliable erfüllt.

Mit diesen Einstellungen und Anpassungen wurden Modelle nach dem vorher beschriebenen Vorgehen trainiert.

### 4.5.2 Bestimmung des besten Modells

Das Ziel war es mit Hilfe dieses Vorgehens relevante Haupt- und Interaktionsterme zu identifizieren, um mit diesen ein einfaches logistisches Regressionsmodell zu trainieren. Dazu wurden mit Hilfe der Verfahren verschiedene Teilmengen an relevanten Variablen festgelegt. Mit diesen Teilmengen wurden anschließend logistische Regressionsmodelle trainiert. Durch Kreuzvalidierung wurde ein bestes Modell bestimmt.

Zunächst wurden relevante Haupt- und Interaktionsterme mit dem KDPI identifiziert. Dazu wurde das minimale $k$ der Folge $\lambda_k$, bei dem der Haupt- bzw. Interaktionsterm mit dem KDPI zum ersten Mal dem jeweiligen Modell hinzugefügt wurde, bestimmt. In allen 50 Bootstrap und für alle Verfahren und Variablen wurde dieser Wert festgehalten. Dieser Wert wird durch $I_v(X_i)^j$ definiert:

$$\begin{aligned} I_v(X_i)^j := \min_k \{k : & X_i \text{ ist im Modell zum Verfahren v in Stichprobe j} \\ & \text{mit Tuningparameter } \lambda_k \text{ enthalten}\} \end{aligned} \tag{4.1}$$

Die Variable $X_i$ kann hier sowohl einem Haupt- als auch Interaktionsterm mit dem KDPI entsprechen. Für jedes der Verfahren wurden die Werte $I_v(X_i)^j$ in tabellarischer Form festgehalten. Ein Auszug aus der resultierenden Tabelle für die Methode GLINTERNET mit einem Lesebeispiel ist in Tabelle 4.2 zu finden.

| Stichprobe | KDPI | GENDER | AGE | ABO_MAT |
|---|---|---|---|---|
| 1 | 1 | 18 | 3 | 46 |
| 2 | 1 | 22 | 3 | 41 |
| 3 | 1 | 20 | 4 | 37 |

**Tab. 4.2:** Ein Auszug aus der resultierenden Tabelle für die Methode GLINTERNET. Bei allen drei Bootstrap Stichproben wurde die Variable KDPI für den ersten Wert $\lambda_1$ (höchste Bestrafung) der Folge dem Modell hinzugefügt. Der Wert $I_{\text{GLINTERNET}}(\text{KDPI})^j$ entspricht also bei allen drei Stichproben eins. Die Variable ABO_MAT wurde erst bei schwächeren Bestrafung, $\lambda_{46}$, $\lambda_{41}$ bzw. $\lambda_{37}$, hinzugefügt. Der Wert $I_{\text{GLINTERNET}}(\text{ABO_MAT})^j$ entspricht also 46, 41 bzw. 37. Bei allen drei Stichproben war damit KDPI die relevanteste und ABO_MAT die am wenigsten relevante Variable.

Relevante Haupt- bzw. Interaktionseffekte wurden bereits bei einer starken Bestrafung, bzw. einem niedrigem $k$ in der Tuning-Parameter Folge dem Modell hinzugefügt. Damit gilt also:

$$I_v(X_i)^j < I_v(X_k)^j \Rightarrow X_i \text{ ist relevanter als } X_k \text{ in Verfahren v bei Stichprobe j}$$

Das Ziel war es mit Hilfe der Tabellen eine Rangliste abzuleiten für die Relevanz von Variablen. Hierzu wurde für alle Variablen der Mittelwert von $I_v(X_i)^j$ aller Verfahren und Stichproben bestimmt:

$$I(X_i) = \frac{1}{4}\frac{1}{50}\sum_{v\in V}\sum_{j=1}^{50} I_v(X_i)^j$$

wobei $V = \{\text{GGLasso, GLINTERNET, Pliable, GAMSEL}\}$.

Aus diesem Überschnitt der Modelle konnte somit jeweils eine Rangliste für die Haupt- und Interaktionsterme gebildet werden, die bei allen Verfahren relevant waren. Hierbei ist zu beachten, dass bei den Verfahren ohne die Möglichkeit kategorielle Variablen zu gruppieren, Pliable und GAMSEL, Kategorien hinzugefügt wurden, sobald eine Ausprägung der Kategorie in das Modell selektiert wurde. Insbesondere bei einer Vielzahl von Ausprägungen, wie beispielsweise bei DIAG_KI, war es damit einfach für eine kategorielle Variable in das Modell aufgenommen zu werden, da lediglich eine Ausprägung der Variable dem Modell hinzugefügt werden musste. Bei GAMSEL wurden beim Überschnitt nur die linearen Effekte betrachtet. Zudem wurde für alle Variablen festgehalten, wie häufig diese den entsprechenden Modellen auch bei geringster Bestrafung nicht hinzugefügt wurden. Es war zu erwarten, dass Variablen, die weniger relevant und somit weiter hinten in der Rangliste waren, häufiger den Modellen nicht hinzugefügt wurden.

Aus den beiden Ranglisten wurden mehrere Teilmengen von Haupt- und Interaktionstermen abgeleitet. Die Bestimmung der Teilmengen erfolgte dabei grafisch. In den Grafiken ist auf der y-Achse der Wert $I(X_i)$ der verschiedenen Variablen angegeben. Die Reihenfolge auf der x-Achse entspricht der Rangfolge der Variablen, die sich aus dem Wert $I(X_i)$ abgeleitet hat. In den Balken ist der Name der jeweiligen Variable enthalten. Der Farbton der Balken gibt an, wie häufig eine Variable nicht selektiert wurde ("Fehlend"). Ein Lesebeispiel um die Aussagen der Abbildung 4.9 des Überschnitts zu verdeutlichen:

Der Haupteffekt der Variable AGE ist sehr relevant, da $I(\text{AGE}) \approx 9$ und somit ist AGE auf Rang 2 der relevanten Hauptterme. Der Interaktionsterm der Variable ABO_MAT mit dem KDPI ist am wenigsten relevant, da $I(\text{ABO_MAT:KDPI}) \approx 44$ (aus Übersichtlichkeit nur ABO_MAT genannt) und somit ist die Interaktion auf dem letzten Platz der relevanten Interaktionsterm. Der dunkle Farbton des Balkens gibt an, dass die Interaktion zudem sehr häufig nicht den Modellen hinzugefügt wurde.

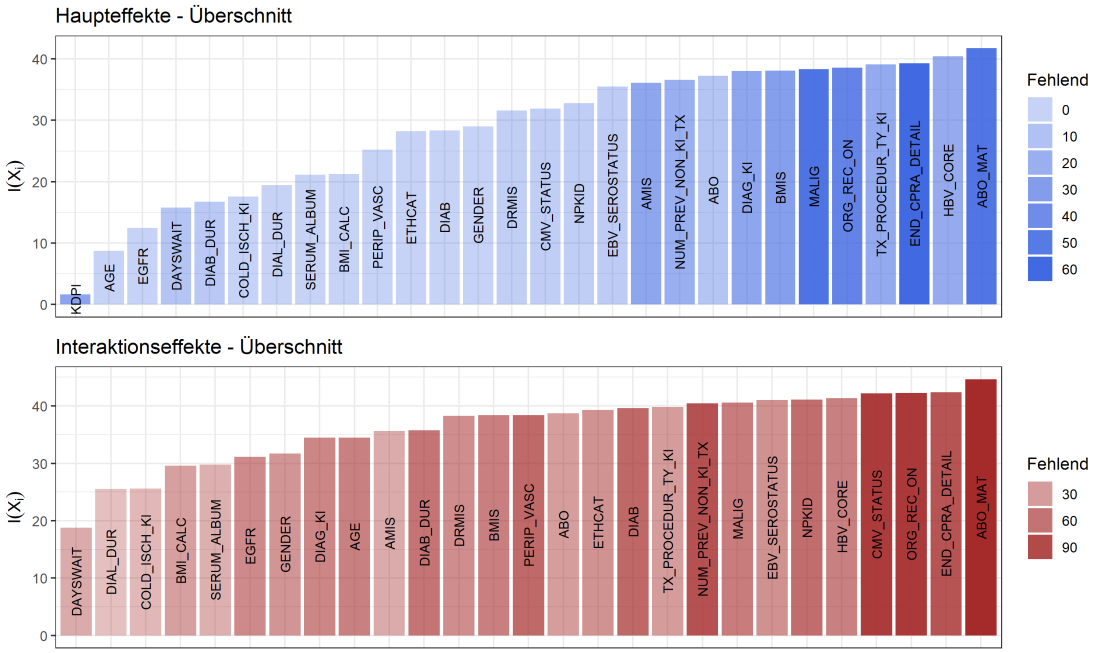


**Abb. 4.9:** Der Überschnitt der Verfahren. Der Farbton gibt an, wie häufig eine Variable bzw. Interaktion den Modellen nicht hinzugefügt wurde. Hierbei wurden die Werte $I_v(X_i)^j$ für alle Verfahren und Stichproben gemittelt. Bei GAMSEL und Pliable wurde eine Kategorie hinzugefügt sobald eine Ausprägung der jeweiligen Kategorie selektiert wurde.

Aus dem Überschnitt in Abbildung 4.9 ergab sich keine klare Teilmenge an relevanten Haupt- und Interaktionstermen. Daher wurden sowohl für die Haupt- als auch die Interaktionsterme jeweils eine konservative und eine erweiterte Teilmenge bestimmt. Die Schwellenwerte für die Teilmengen wurden dabei auf Basis von Auffälligkeiten in den Grafiken bestimmt.

Die konservative Teilmenge bei den Haupteffekten besteht aus: KDPI, AGE, EGFR, DAYSWAIT, DIAB_DUR, COLD_ISCH_KI, DIAL_DUR, SERUM_ALBUM und BMI_CALC. Zur nächsten Variable ist dort ein größerer Sprung in der Rangliste zu sehen. Die erweiterte Teilmenge enthält zusätzlich PERIP_VASC, ETHCAT, DIAB, GENDER, DRMIS, CMV_STATUS, NPKID und EBV_SEROSTATUS. Danach wurden viele Variablen in den vier Verfahren nicht den Modellen hinzugefügt.

Bei den Interaktionseffekten wurde als konservative Teilmenge DAYSWAIT, DIAL_DUR und COLD_ISCH_KI gewählt. Zur nächsten Variable ist hier ein größerer Sprung zu sehen. Für die erweiterte Teilmenge wurde zusätzlich noch BMI_CALC und SERUM_ALBUM hinzugefügt. Anschließend wurden viele Interaktionen in den vier Verfahren nicht den Modellen hinzugefügt.

Die Methode GAMSEL bietet die Möglichkeit auch nicht-lineare Effekte zu modellieren. Hierzu wurde eine separate Rangliste für die nicht-linearen Effekte erstellt. Diese Rangliste ist in Abbildung 4.10 zu sehen. Insgesamt wurden die nicht-linearen Komponenten als wenig relevant geschätzt und erst bei hohen Werten $k$ den Modellen hinzugefügt. Zudem wurden bei fast allen Variablen bei mindestens der Hälfte der Stichproben auch bei der geringsten Bestrafung kein nicht-linearer Effekt entdeckt. Eine Ausnahme war hier lediglich die Variable AGE. Für diese wurde bei allen Stichproben ein nicht-linearer Effekt hinzugefügt.

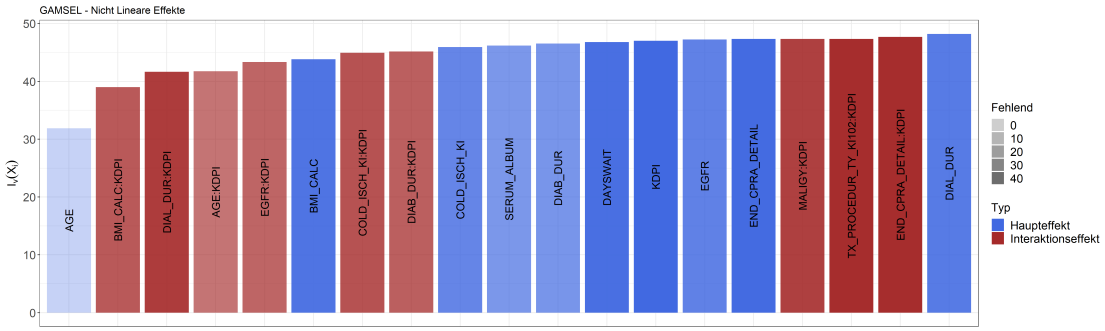


**Abb. 4.10:** Die Rangliste der nicht-linearen Effekte von GAMSEL. Der Farbton gibt an, wie häufig eine Variable bzw. eine Interaktionen einem Modell nicht hinzugefügt wurde.

Durch den Tuning-Parameter $\gamma$ in der Methode können ein nicht-linearer oder linearer Effekt bevorzugt werden. Hierbei wurde der Parameter auf dem Standardwert $\gamma = 0.4$ belassen. Nach den Autoren wird hier durch ein linearer Effekt für eine Variable leicht bevorzugt, der moderate Wert erlaubt es aber dennoch nicht-linearen Effekten in das Modell aufgenommen zu werden (Chouldechova & Hastie 2015). Es wurden daher nur nicht-lineare Effekte mit der Variable AGE im nächsten Schritt betrachtet.

Mittels der beiden Teilmengen der relevanten Haupteffekte, der beiden Teilmengen der relevanten Interaktionseffekte mit dem KDPI und dem nicht-linearen Effekt der Variable AGE wurden logistische Regressionsmodelle berechnet. Um die Unterschiede zwischen den Teilmengen festzustellen, wurden für alle Kombinationen aus diesen Teilmengen Modelle trainiert. Als zusätzliche Vergleichsmodelle wurden noch Modelle ohne Variablenselektion mit bzw. ohne Interaktionsterme erstellt. Die Modelle und die Farblegenden , die für die nächsten Grafiken benötigt wird, sind in Abbildung 4.11 dargestellt.

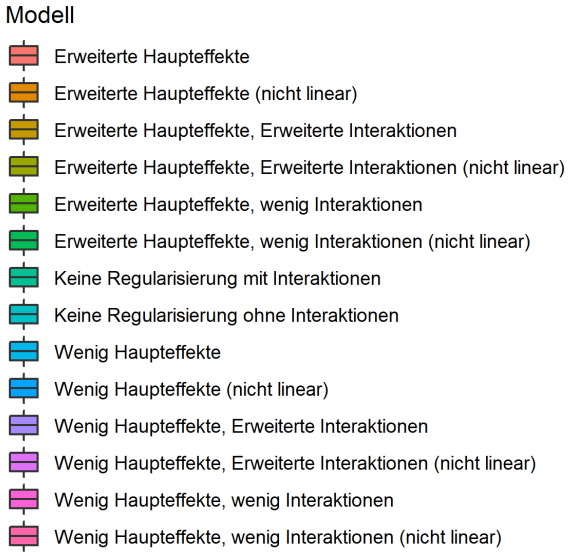


**Abb. 4.11:** Die Modelle und ihre Farblegenden. Mit diesen Modellen wurden logistische Regressionsmodelle trainiert und mittels Kreuzvalidierung verglichen.

Für alle Modelle wurde der Kreuzvalidierungsfehler bestimmt. Um dabei sowohl die Varianz als auch den Bias der Modelle zu berücksichtigen, wurde der Kreuzvalidierungsfehler für die zwei üblichen Parameter $K = 5$ und $K = 10$ berechnet. Die Kreuzvalidierung wurde für jeweils 100 verschiedene Einteilung des Datensatzes iteriert.

Beim Anwendungsfall ist die Vorhersage der Wahrscheinlichkeit für Versterben und erneutem Nierenversagen des Empfängers von Interesse. Zudem ist die Zielgröße einseitig verteilt, lediglich 6,3% der Patienten sind innerhalb eines Jahres verstorben oder hatten Nierenversagen. Als Fehlermaß wurde daher der BSS gewählt. Zur Berechnung des BSS wird ein Referenzmodell benötigt mit dem die anderen Modelle verglichen werden können. In diesem Fall prognostizierte das Referenzmodell für alle Beobachtungen die Klassenverteilung. Für jeden Patienten wurde unabhängig seiner Eigenschaften eine Wahrscheinlichkeit für Versterben oder erneutem Nierenversagen von 6,3% im Referenzmodell vorhergesagt.

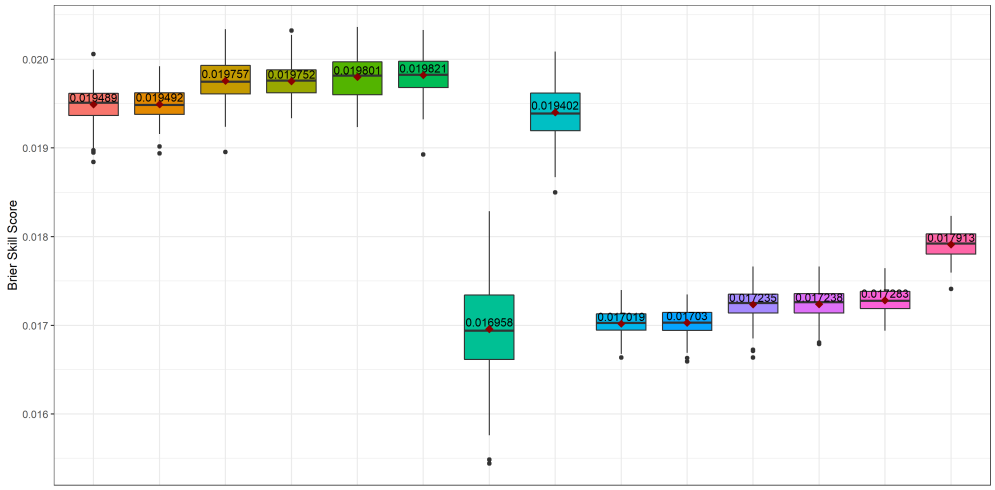


**Abb. 4.12:** Der Kreuzvalidierungsfehler der Modelle mit K=5. Die Kreuzvalidierung wurde für 100 verschiedene Einteilung des Datensatzes iteriert. Der Wert in den Boxplots entspricht dem Mittelwert des BSS des jeweiligen Modells. Die Farblegenden sind aus Abbildung 4.11 zu entnehmen.

In Abbildung 4.12 ist die Verteilung des Kreuzvalidierungsfehlers mit $K = 5$ für die Iterationen aller Verfahren zu sehen. Mit Hilfe dieser Grafik wurden die Modelle miteinander verglichen. Hierbei ist zu beachten, dass beim BSS ein höherer Wert eine bessere Performance bedeutet.

Die Modelle mit der konservativen Teilmenge an Haupteffekten schnitten allesamt schlecht ab und waren nur geringfügig besser als das Modell ohne Variablenselektion mit Interaktionen. Die vielen Variablen in diesem Modell führten zu einer hohen Varianz, die im Boxplot des Modells zu sehen ist.

Die Modelle mit mehr Haupteffekten schnitten besser ab, dennoch waren auch diese nur geringfügig besser als ein Modell ohne Variablenselektion und ohne Interaktionen. Die Interaktionen mit dem KDPI führten insgesamt bei allen Modellen nur zu einer geringen Verbesserung. Erweiterte Haupteffekte und konservative Interaktionen lieferten insgesamt die beiden besten Modelle. Der Unterschied zwischen dem Modell mit linearen und nicht-linearen Effekt der Variable AGE war nur marginal. Aufgrund der besseren Erklärbarkeit des linearen Modells wurde insgesamt das Modell mit erweiterten Haupteffekten und konservativen Interaktionen als das beste Modell für $K = 5$ bestimmt.

Die Verteilungen der Kreuzvalidierungsfehler für $K = 10$ zeigte ein ähnliches Bild (Abbildung 4.13). Auch mit dieser Parameterwahl waren die Modelle mit den erweiterten Haupteffekten besser als die Modelle mit der konservativen Teilmenge. Zudem konnten auch hier die Interaktionen die Modelle nur geringfügig verbessern. Das Modell ohne Variablenselektion mit Interaktionen war sogar etwas besser als die Modelle mit

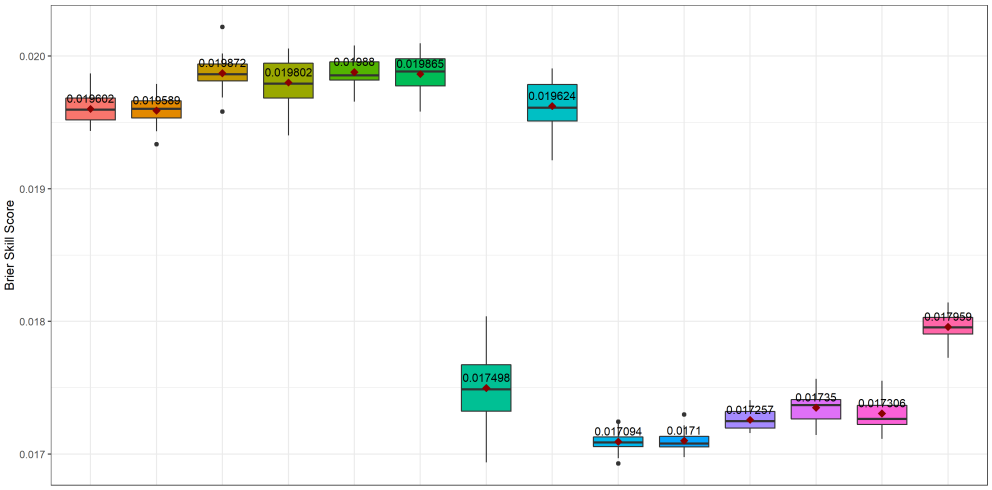


**Abb. 4.13:** Der Kreuzvalidierungsfehler der Modelle mit K=10. Die Kreuzvalidierung wurde für 100 verschiedene Einteilung des Datensatzes iteriert. Der Wert in den Boxplots entspricht dem Mittelwert des BSS des jeweiligen Modells. Die Farblegenden sind aus Abbildung 4.11 zu entnehmen.

wenigen Haupteffekten. Der höhere Parameter $K = 10$ führte dazu, dass mit mehr Daten trainiert wurde und somit die hohe Varianz des Modells weniger ins Gewicht fiel.

Insgesamt zeigte das Modell mit erweiterten Haupteffekten und konservativen Interaktionen die beste Performance. Damit wurden die relevanten Haupteffekte: KDPI, AGE, EGFR, DAYSWAIT, DIAB_DUR, COLD_ISCH_KI, BMI_CALC, DIAL_DUR, SERUM_ALBUM, PERIP_VASC, ETHCAT, DIAB, GENDER, DRMIS, CMV_STATUS, NPKID und EBV_SEROSTATUS und die relevanten Interaktionseffekte mit dem KDPI: DAYSWAIT, DIAL_DUR und COLD_ISCH_KI festgelegt. Für diese Variablen und Interaktionen wurde ein finales logistisches Regressionsmodell bestimmt und persistiert.

## 4.6 Evaluation

Mit Hilfe des Selektionsprozesses konnte die Anzahl Variablen von 28 Haupttermen und entsprechend 27 Interaktionstermen mit dem KDPI reduziert werden auf 17 Hauptterme und 3 Interaktionsterme. Es wurden vor allem stetige Variablen und Interaktionen mit diesen selektiert, daher konnte die Anzahl an zu berechnenden Koeffizienten sogar von 161 auf 29 reduziert werden (ohne Intercept). Durch die gute Erklärbarkeit des logistischen Regressionsmodells konnten die Einflüsse von Variablen und insbesondere der selektierten Interaktionen mit dem KDPI im nächsten Schritt in einem Dashboard dargestellt werden.

Zur Evaluation der einzelnen Variablen wurden in der Tabelle 4.3 die Koeffizienten und p-Werte festgehalten. Bei den Koeffizienten des Modells fielen einige Variablen auf:

- **EGFR_CKDEPI_TRR_PRETX**: Die Estimated Glomerular Filtration Rate (GFR) gibt die Filtrationsrate der Niere an. Je niedriger Wert, desto kritischer ist der Status der Niere (*Estimated Glomerular Filtration Rate (eGFR)* 2020). Es ist überraschend, dass der Koeffizient von GFR negativ ist. Eine höhere Filtrationsrate führt somit zu einer schlechteren Prognose des Patienten.
- **CMV_STATUS**: Diese Variable gibt an, ob eine Cytomegalovirus Erkrankung vorliegt. Es ist überraschend, dass eine positive Infektion zu einer besseren Prognose des Empfängers führt.
- **EBV_SEROSTATUS**: Diese Variable gibt an, ob eine Infektion mit dem Epstein-Barr-Virus vorliegt. Auch bei dieser Variable führt eine positive Diagnose zu einer besseren Prognose des Empfängers.

Aus den p-Werten lies sich ablesen, dass die meisten Variablen signifikant zu einem Signifikanzniveau von 95% waren. Einige Ausprägungen von kategoriellen Variablen fielen hierbei unter dieses Niveau. Die Variable DIAB_DUR fiel durch einen sehr hohen p-Wert auf, obwohl sie im Überschnitt aus Abbildung 4.9 auf Rang 5 der relevanten Haupteffekte zu finden war. Zwei der gefunden Interaktionen fielen ebenfalls unter das übliche 95% Signifikanzniveau.

| Variable | Koeffizient | p-Wert |
|---|---|---|
| Intercept | -3,742288 | < 2e-16 |
| KDPI | 0,510917 | 0,00891 |
| AGE | 0,014714 | < 2e-16 |
| DAYSWAIT_CHRON_KI | -0,000126 | 0,016232 |
| EGFR_CKDEPI_TRR_PRETX | 0,018702 | < 2e-16 |
| DIAB_DUR | 2e-05 | 0,992963 |
| COLD_ISCH_KI | 0,008349 | 0,059593 |
| BMI_CALC | 0,011925 | 0,000771 |
| DIAL_DUR | 0,023766 | 0,043188 |
| TOT_SERUM_ALBUM | -0,233548 | < 2e-16 |
| PERIP_VASCPositiv | 0,265056 | < 2e-16 |
| GENDERMännlich | 0,118798 | 0,002008 |
| ETHCATSchwarz | -0,07066 | 0,122878 |
| ETHCATHispanisch | -0,354479 | < 2e-16 |
| ETHCATAsiatisch | -0,320396 | 0,000159 |
| ETHCATAlaskanisch | -0,184837 | 0,319184 |
| ETHCATHawaiianisch | -0,446792 | 0,172722 |
| ETHCATGemischt | -0,309643 | 0,154897 |
| DIABTypI | 0,209305 | 0,104134 |
| DIABTypII | 0,2042 | 0,000375 |
| DIABAndererTyp | 0,040281 | 0,826181 |
| NPKID | 0,16472 | 0,000977 |
| DRMIS1 | 0,039392 | 0,478454 |
| DRMIS2 | 0,154885 | 0,006956 |
| CMV_STATUSPositiv | -0,073148 | 0,080243 |
| EBV_SEROSTATUSPositiv | -0,193789 | 0,003028 |
| KDPI:DAYSWAIT_CHRON_KI | 0,000133 | 0,146167 |
| KDPI:COLD_ISCH_KI | 0,011683 | 0,134748 |
| KDPI:DIAL_DUR | 0,056977 | 0,007718 |

**Tab. 4.3:** Die Koeffizienten und die p-Werte des finalen Modells.

## 4.7 Deployment

Das im letzten Schritt bestimmte Modell wurde mittels verschiedener interaktiver Grafiken und Metriken den Anwendern verfügbar gemacht. Das Dashboard wurde mit Hilfe von R-Shiny erstellt. R-Shiny bietet die Möglichkeit interaktive Web-Anwendungen zu erstellen. Die App setzt sich zusammen aus einem User Interface, das die Benutzeroberfläche für den Nutzer zur Verfügung stellt und einem Server, auf dem die eigentliche Anwendung läuft. Der Server benötigt sowohl einen Teil des Datensatzes als auch das persistierte Modell, um das Dashboard zu generieren.

Bei dem interaktiven Dashboard war es wichtig, dass die Grafiken und Metriken mit entsprechenden Erklärungstexten auch ohne statistisches Fachwissen einfach zu verstehen sind. Hierbei werden sowohl ein Datenverständnis durch deskriptive Statistik vermittelt, als auch die Erkenntnisse, die durch das Modell gewonnen wurden, verständlich gemacht. Die Interaktionen mit dem KDPI sind durch Interaktionsplots ebenfalls anschaulich dargestellt.

Die App wurde in drei Bereiche unterteilt. Mittels einer Sidebar hat der Anwender die Möglichkeit zwischen den verschiedenen Bereichen, wie in Abbildung 4.14 zu sehen, zu wechseln. Im folgenden werden die drei Bereiche: Deskriptive Statistik, Vorhersagemodell und Interaktionen mit KDPI erläutert.

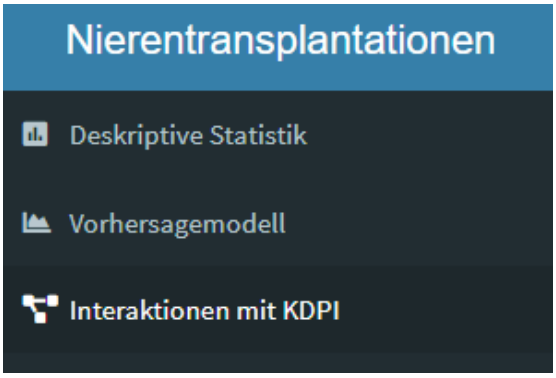


**Abb. 4.14:** Die Wahlmöglichkeiten des Anwenders in der Sidebar des Dashboards. In der Kategorie Deskriptive Statistik wird ein Datenverständnis mittels Histogrammen vermittelt. Im Bereich Vorhersagemodell ist das persistierte Modell dargestellt. Unter dem Menüpunkt Interaktionen mit KDPI sind Interaktionsplots zu den selektieren Interaktionen mit dem KDPI dargestellt.

### 4.7.1 Deskriptive Statistik

**Abb. 4.15:** Der Anwender kann im Bereich Deskriptive Statistik die Verteilungen der Variablen mit Histogrammen betrachten. Zudem ist festgehalten wo die Daten herkommen und welche Filter gesetzt wurden.

In der Kategorie Deskriptive Statistik hat der Anwender die Möglichkeit Histogramme über die verschiedenen Variablen zu betrachten. Hierbei wurde zwischen stetigen und kategoriellen Variablen unterschieden. Es stehen alle Variablen aus dem finalen Modell zur Auswahl. Die Verteilungen beziehen sich immer auf den entsprechend eingeschränkten Datensatz (siehe Kapitel 4.4). Mit Hilfe dieser Grafiken kann sich der Anwender einen ersten Eindruck über die Daten verschaffen und ein grundlegendes Verständnis für die Variablen entwickeln.

Des Weiteren wird in diesem Bereich die Herkunft des Datensatzes beschrieben. Zudem wird erläutert, wie das das Patientenkollektiv eingeschränkt wurde. Mit Hilfe dieser Erklärung kann der Anwender die Erkenntnisse besser beurteilen.

### 4.7.2 Vorhersagemodell

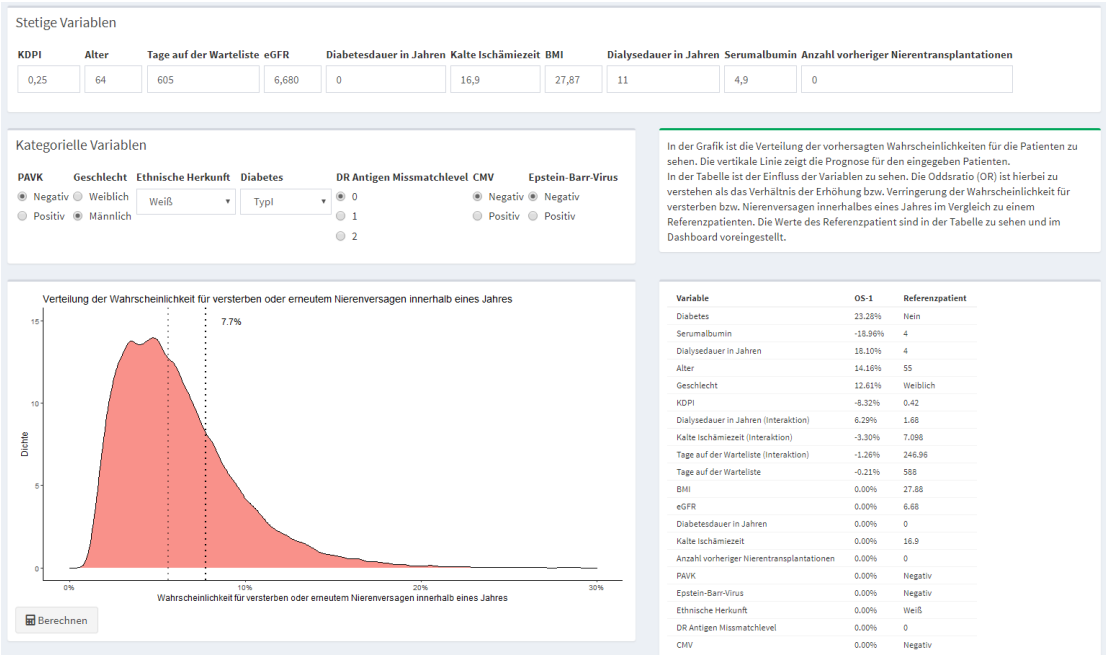


**Abb. 4.16:** Der Anwender kann durch Eingabe von Merkmalen eines Patienten und dem KDPI eine Prognose für Versterben bzw. Nierenversagen innerhalb eines Jahr erstellen. Des Weiteren erhält er detaillierte Informationen über den Einfluss der einzelnen Variablen auf die Prognose.

In der Kategorie Vorhersagemodell ist das finale Modell visualisiert. Der Anwender hat die Möglichkeit die Parameter des potentiellen Empfängers zu variieren, wobei diese standardmäßig auf den Median bei stetigen bzw. auf die Referenzkategorie bei den kategoriellen Variablen voreingestellt sind. Durch Anklicken von "Berechnen" unterhalb der Grafik wird die Vorhersage berechnet und die Grafik und Tabelle aktualisiert.

In der Grafik ist mittels Kerndichteschätzung die Verteilung der vorhergesagten Wahrscheinlichkeiten für alle Patienten im Datensatz zu sehen. Die dunkle, vertikale, gestrichelte Linie stellt die vorhergesagte Sterbewahrscheinlichkeit innerhalb eines Jahres für den Empfänger mit den eingestellten Parametern dar. Die hellere Linie zeigt die Vorhersage für einen Referenzpatienten. Dieser Referenzpatient hat für die stetigen Merkmale den Median und für die kategoriellen Variablen die Referenzkategorie als Eigenschaft. Mit Hilfe der Grafik kann der Anwender einschätzen wie wahrscheinlich Versterben bzw. Nierenversagen innerhalb eines Jahres nach Nierentransplantation ist. Durch die Darstellung der Gesamtpopulation und des Referenzpatienten lässt sich die Wahrscheinlichkeit besser vergleichen.

In der Tabelle ist der Einfluss der verschiedenen Variablen zu sehen. Hierbei wird der eingestellte Empfänger durch Klicken auf "Berechnen" verglichen mit dem Referenzpatienten. Für jede Variable wird der Effekt als die relative Änderung der Oddsratio-1 berechnet:

- **Stetige Variable:** Die Berechnung des Effekts $\text{Eff}_{x_j}$ im Vergleich zum Referenzpatient für eine stetige Variable $X_j$ mit Realisierung $x_j$ und Median $Med_{X_j}$ erfolgt mit:

$$\text{Eff}_{x_j} = \exp\left(\beta_j \cdot (x_j - Med_{X_j})\right) - 1$$

- **Kategorielle Variablen:** Für eine kategorielle Variable $X_j$ wurde für jede Ausprägung $x_j = A$, außer der Referenzkategorie $x_j = \text{ref}$, ein Koeffizient $\beta_{j_A}$ berechnet. Der Effekt $\text{Eff}_{x_j}$ einer kategoriellen Variable im Vergleich zum Referenzpatient lässt sich dann berechnen mit:

$$\text{Eff}_{x_j} = \begin{cases} 0 & \text{falls } x = \text{ref} \\ \exp\left(\beta_{j_A}\right) - 1 & \text{sonst} \end{cases}$$

- **Interaktionen mit KDPI:** Für den Interaktionseffekt $\text{Eff}_{x_i x_j}$ der Variable $X_i$ mit Realisierung $x_i$ mit dem KDPI mit der Realisierung $x_j$ im Vergleich zum Referenzpatient gilt:

$$\text{Eff}_{x_i x_j} = \exp\left(\theta_{ij} \cdot (x_j x_i - (Med_{X_i} \cdot Med_{\text{KDPI}}))\right) - 1$$

Durch die Tabelle kann der Anwender einschätzen, welche Faktoren und Interaktionen im Vergleich zum Referenzpatient besonders einflussreich auf die Prognose sind. Ein Wert über null gibt dabei eine Erhöhung und ein Wert unter null eine Verringerung des Risikos für Versterben oder Nierenversagen innerhalb eines Jahres nach Nierentransplantation an.

### 4.7.3 Interaktionen mit KDPI

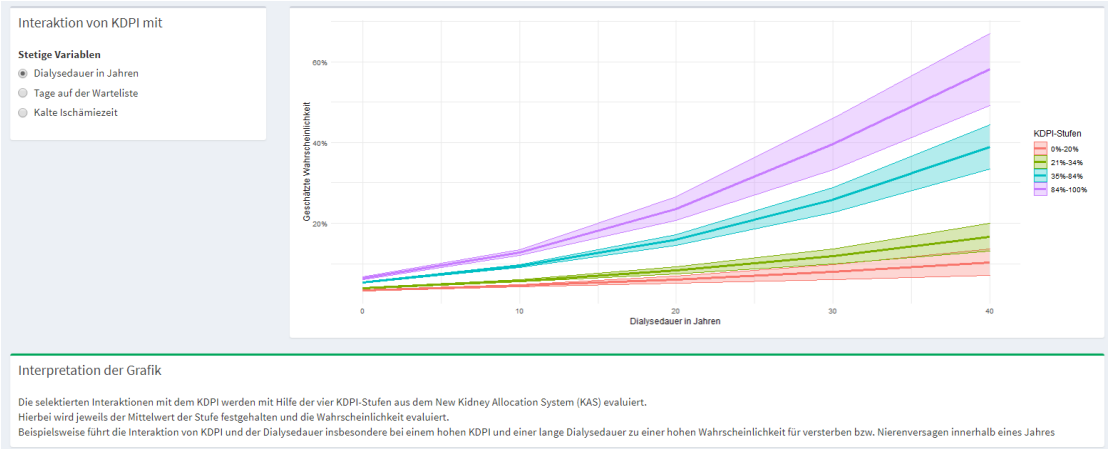


**Abb. 4.17:** Der Anwender kann den Einfluss der selektieren Interaktionen zwischen dem KDPI und den einzelnen Variablen untersuchen. Hierbei wird für jede Stufe des KDPI aus dem Allokationsprozess der Mittelwert festgehalten und die Prognose berechnet. Die Flächen um die Linien geben dabei die Konfidenzintervalle an.

Im Bereich Interaktionen mit KDPI werden die selektierten Interaktionen mit dem KDPI im finalen Modell dargestellt. Hierbei wird der KDPI in die vier Stufen 0%-20%, 21%-34%, 35%-84% und 85%-100%, die beim KAS eingesetzt werden, eingeteilt. Für jede Stufe wird der Mittelwert des KDPI der jeweiligen Stufe festgehalten. In der Grafik ist die Entwicklung der Vorhersage für die verschiedenen Stufen des KDPI bei Veränderung der interagierenden Variable dargestellt.

In Abbildung 4.17 sieht man beispielhaft die Interaktion zwischen der Variable "Dialysedauer in Jahren" und dem KDPI. Die Interaktion zwischen den Variablen erhöht die Wahrscheinlichkeit für Versterben bzw. Nierenversagen innerhalb eines Jahres. Insbesondere Nieren aus der letzten KDPI Stufe haben dadurch einen starken Einfluss auf die Prognose für Patienten, die schon lange die Dialyse erhalten. Die Flächen um die Linien zeigen dabei die Konfidenzintervalle der Vorhersage.

# 5 Systematischer Vergleich der Verfahren

In diesem Kapitel werden die Verfahren systematisch miteinander verglichen. Hierzu wurden zum einen die Unterschiede der Variablenselektion im Anwendungsfall des vorherigen Kapitels untersucht und zum anderen die Methoden in einer Simulationsstudie geprüft.

## 5.1 Unterschiede im Anwendungsfall

Um die Ergebnisse der Modelle im Anwendungsfall gegenüberzustellen, wurden für jedes Verfahren Ranglisten erstellt. Die Methodik ist hierbei ähnlich dem Vorgehen in Kapitel 4. Dafür wurde für jedes Verfahren für die Haupt- und Interaktionsterme jeweils eine Rangliste erstellt. Für jede Variable $X_i$ wurde in jedem Verfahren $v$ dafür der folgende Wert berechnet:

$$I_v(X_i) = \frac{1}{50} \sum_{j=1}^{50} I_v(X_i)^j$$

wobei $v \in \{\text{GGLasso, GLINTERNET, Pliable, GAMSEL}\}$ und $I_v(X_i)^j$ definiert wie in Definition 4.1.

Aus diesen Werten wurden Grafiken abgeleitet. Mit Hilfe der Grafiken wurden die relevanten Haupt- und Interaktionsterme miteinander verglichen und besondere Auffälligkeiten in den Verfahren festgestellt. Ein Lesebeispiel für die Haupteffekte von GGLasso in Abbildung 5.1 verdeutlicht die Aussagen der Grafiken:

Der Haupteffekt der Variable AGE wurde bei GGLasso ungefähr bei $I_{GGLasso}(AGE) \approx 15$ den Modellen hinzugefügt. Damit war diese Variable der drittwichtigste Haupteffekt in dieser Methode. Der helle Farbton zeigt, dass die Variable immer im Modell enthalten war.

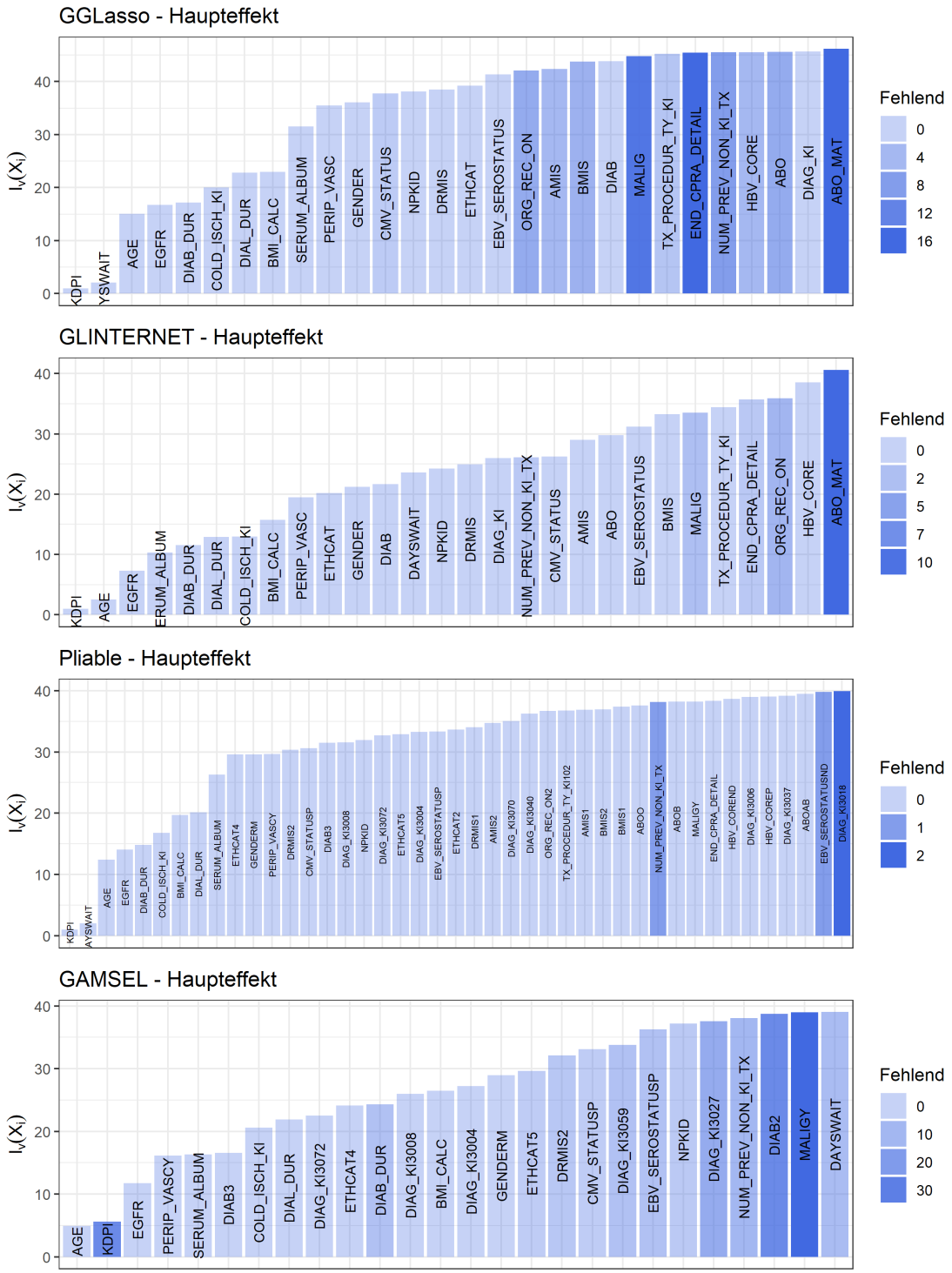


**Abb. 5.1:** Die Rangliste der Haupteffekte der Verfahren. Der Farbton gibt an, wie häufig eine Variable einem Modell nicht hinzugefügt wurde. Bei den Methoden ohne die Möglichkeit kategorielle Variablen zu gruppieren, Pliable und GAMSEL, wurden aus Übersichtlichkeit nur Variablen mit einem Durchschnittswert von unter 40 betrachtet.

Zunächst wurden die Ranglisten der Haupteffekte der vier Methoden mit Hilfe der Abbildung 5.1 miteinander verglichen. Der KDPI war in den Methoden Pliable und GGLasso nicht im Strafterm enthalten und konnte somit nicht aus den Modellen entfernt werden. Es ist zu beachten, dass die Modelle mit den Residuen eines linearen Modells (siehe 4.4.3) statt dem KDPI trainiert wurden, um mit dem Confounding umzugehen. Trotz dieser Einschränkung ist der KDPI auch bei GAMSEL und GLINTERNET hochrelevant und an erster bzw. zweiter Stelle in der Rangliste der Haupteffekte zu finden.

Es ließen sich viele Überschneidungen in den Ranglisten der verschiedenen Methoden finden. Insbesondere stetige Merkmale, wie BMI, AGE oder EGFR waren bei allen Verfahren weit vorne in der Rangliste und somit relevante Variablen. Kategorielle Variablen waren in den Ranglisten tendenziell weiter hinten und damit insgesamt weniger relevant. Vereinzelt waren einzelne Ausprägungen von Kategorien, wie beispielsweise ETHCAT3 oder DIAB3, bei den Verfahren ohne die Möglichkeit kategorielle Variablen zu gruppieren, GAMSEL und Pliable, weiter vorne in der Rangliste zu finden.

Bei der Variable DAYSWAIT fiel auf, dass diese bei den beiden Verfahren Pliable und GGLasso hochrelevant ist und an zweiter Stelle der Rangliste, jedoch bei GLINTERNET und GAMSEL als weniger relevant geschätzt wurde. Generell selektierten Pliable und GGLasso zunächst alle stetigen Variablen mit Ausnahme von NPKID und NUM_PREV_NON_KI_TX. Diese Variablen beschreiben wieviele Nieren- bzw. Nicht-Nierentransplantationen der Patient bereits hatte und haben entsprechend wenige Ausprägungen. Hier könnte ein Problem vorliegen, dass die beiden Verfahren die Relevanz von Variablen mit wenigen Ausprägungen unterschätzen.

Die Variable ABO_MAT, die für die Abweichung der Blutgruppen zwischen Empfänger und Spender steht, wurde bei allen Verfahren als wenig relevant geschätzt. Eine mögliche fachliche Erklärung ist, dass durch präventive Therapien und Methoden unterschiedliche Blutgruppen bei Nierentransplantation mittlerweile weitestgehend kein Problem mehr sind (Zschiedrich *et al.* 2015).

Bei GAMSEL war zudem auffallend, dass der KDPI zwar an zweiter Stelle der Rangliste, jedoch bei 33 der 50 Bootstrap Stichproben nicht im Modell enthalten war. Dies spricht für eine hohe Varianz zwischen den Modellen von GAMSEL. Zudem wurden im Vergleich zu Pliable vermehrt einzelne Ausprägungen von kategoriellen Variablen gewählt.

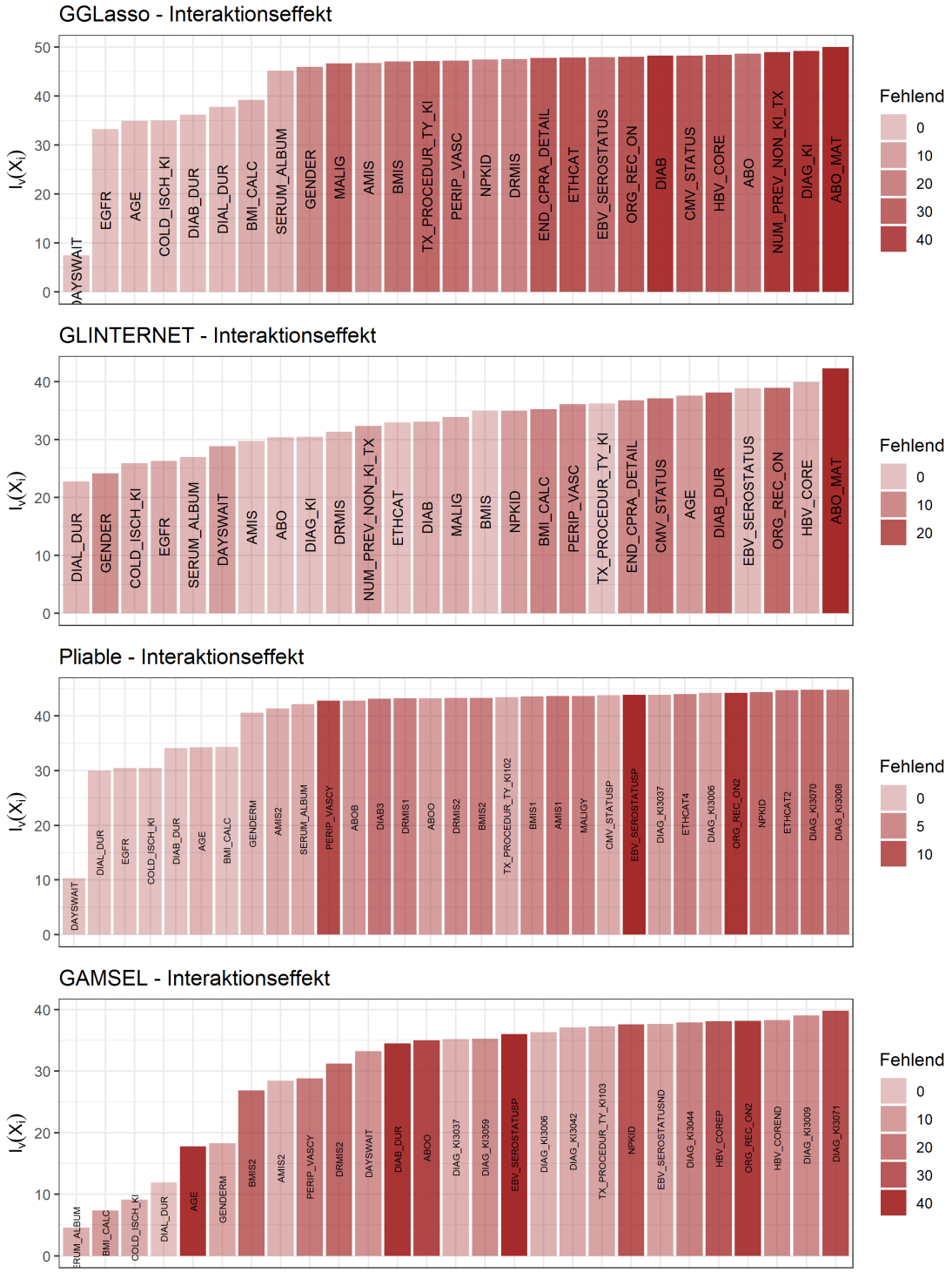


**Abb. 5.2:** Die Rangliste der Interaktionseffekte der Verfahren. Der Farbton gibt an, wie häufig eine Interaktion einem Modell nicht hinzugefügt wurde. Bei den Methoden ohne die Möglichkeit kategorielle Variablen zu gruppieren, Pliable und GAMSEL, wurden aus Übersichtlichkeit nur Variablen mit einem Durchschnittswert von unter 45 bzw. 40 betrachtet.

Als nächstes wurden die Ranglisten der Interaktionseffekte mit Hilfe der Abbildung 5.2 miteinander verglichen. Bei GLINTERNET und Pliable wurden die Interaktionen mit dem KDPI durchschnittlich erst bei einem höheren $I_v(X_i)$ als die Hautpeffekte in das Modell aufgenommen. Die Haupteffekte mussten durch die starke Hierarchiebedingung zwangsweise bei einem niedrigeren oder gleichen Tuning-Parameter im Vergleich zu den Interaktionseffekten in die Modelle aufgenommen werden.

Bei GGLasso wurden die Interaktionen ebenfalls erst bei einem höheren $I_v(X_i)$ den Modellen hinzugefügt. Da bei GGLasso keine starke Hierarchiebedingung besteht, wurden die Interaktionen hierbei aufgrund geringerer Relevanz erst später in die Modelle aufgenommen.

Auch die Interaktion zwischen dem KDPI und DAYSWAIT wurden bei GGLasso und Pliable als mit Abstand am relevantesten eingeschätzt. Insgesamt zeigte sich auch bei den Interaktionen bei den beiden Verfahren erneute eine starke Präferenz von stetigen Variablen mit vielen Ausprägungen.

Bei der Methode GAMSEL fiel auch bei den Interaktionen ein ungewöhnliches Muster bei den Variablen, die nicht den Modellen hinzugefügt wurden, auf. Beispielsweise tauchte die Interaktion von AGE mit dem KDPI durchschnittlich zwar schon ungefähr bei $\lambda_{17}$ auf und war somit an fünfter Stelle in der Rangliste, war jedoch bei 41 der 50 Stichproben nicht im Modell enthalten.

Die Interaktion der Variable ABO_MAT mit dem KDPI ist ebenso wie der Haupteffekt bei allen Methoden als wenig relevant geschätzt worden. Dies spricht für die Annahme, dass verschiedene Blutgruppen bei Nierentransplantationen mittlerweile weitestgehend kein Problem mehr sind.

Insgesamt zeigte sich im Anwendungsfall bei allen Verfahren eine Präferenz der stetigen Merkmale. Insbesondere die Methoden GGLasso und Pliable selektierten überwiegend stetige Merkmale mit vielen Ausprägungen. GAMSEL zeigte eine sehr hohe Varianz zwischen den Modellen der verschiedenen Bootstrap Stichproben.

Ein weiterer großer Unterschied zwischen den Verfahren bestand zwischen den Laufzeiten (Tabelle 5.1). Vor allem die Berechnung der Modelle mit GGLasso dauert im Vergleich zu den anderen erheblich länger. Pliable konnte hierbei durch eine kurze Laufzeit von nur wenigen Sekunden überzeugen.

| Verfahren | Durchschnittliche Laufzeit |
|---|---|
| GLINTERNET | 418,95s |
| GGLasso | 5474,36s |
| GAMSEL | 47,24s |
| Pliable | 6,95s |

**Tab. 5.1:** Die durchschnittliche Laufzeit der Verfahren für eine Iteration. Die Berechnung von GGLasso dauerte erheblich länger als die Berechnung der anderen Verfahren.

## 5.2 Simulationsstudie

Die Simulationsstudie und die Darstellungen orientieren sich am Artikel *Do little Interactions get lost in dark random forests?* von Wright et al. In der Studie untersuchte Wright die Möglichkeit mit Random Forests paarweise Interaktionen zwischen Variablen zu identifizieren. Hierzu wurde ein Datensatz mit vielen Variablen simuliert, wobei lediglich zwischen zwei Variablen tatsächlich eine Interaktion bestand. Mit verschiedenen Effektstärken in der Simulation und Parametereinstellungen der Random Forests wurde die Effizienz der Identifikation festgestellt (Wright *et al.* 2016).

### 5.2.1 Aufbau der Simulationsstudie

In dieser Simulationsstudie wurde untersucht, wie effizient die vier Regressionsmodelle Haupt- und Interaktionsterme mit einer ausgewählten Variable bei einer Vielzahl von Störvariablen identifizieren können. Dazu wurden zwei stetige Variablen $X_1$ und $X_2$ und die Interaktion zwischen diesen simuliert. Die hier untersuchten Regressionsmodelle unterscheiden sich unter anderem durch die Möglichkeit kategorielle Variablen zu gruppieren. Um den Unterschied dieser zwei Arten von Verfahren zu untersuchen wurde, zusätzliche eine kategorielle Variable $X_3$ und die Interaktion zwischen $X_1$ und $X_3$ simuliert. Die Simulation des linearen Prädiktors hatte die Form:

$$\begin{aligned} Y_{lin} = \quad & \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \theta_1 X_1 X_2 + \\ & \beta_{3A} X_{3A} + \beta_{3B} X_{3B} + \beta_{3C} X_{3C} + \beta_{3D} X_{3D} + \beta_{3E} X_{3E} + \\ & \theta_{2A} X_1 X_{3A} + \theta_{2B} X_1 X_{3B} + \theta_{2C} X_1 X_{3C} + \theta_{2D} X_1 X_{3D} + \theta_{2E} X_1 X_{3E} + \varepsilon \end{aligned}$$

Die stetigen Variablen $X_1$ und $X_2$ und der nicht reduzierbare Fehler $\varepsilon$ waren $\mathcal{N}(0,1)$ standardnormalverteilt. Die kategorielle Variable $X_3$ hatte fünf Ausprägungen $\{A,B,C,D,E\}$, wobei jede Kategorie mit einer Wahrscheinlichkeit von 20% auftrat. Die Variable $X_3$

wurde in Dummy Variablen aufgelöst und jede Ausprägung erhielt einen Koeffizienten. Die Wahl der Koeffizienten geschah iterativ durch Versuche, bis eine geeignete Wahl gefunden war. Die Werte für die Koeffizienten sind in Tabelle 5.2 zu sehen.

| Parameter | Werte |
|---|---|
| $\beta_0$ | -1 |
| $\beta_1, \beta_2, \theta_1$ | Kein Effekt: 0, Schwacher Effekt: 0,2, Starker Effekt: 0,4 |
| $\beta_{3A}$,$\beta_{3B}$,$\beta_{3C}$,$\beta_{3D}$,$\beta_{3E}$, $\theta_{2A}$,$\theta_{2B}\theta_{2C}\theta_{2D}\theta_{2E}$ | Kein Effekt: 0, Schwacher Effekt: 0,5, Starker Effekt: 1 |

**Tab. 5.2:** Die Werte der Koeffizienten des simulierten Modells. Es wurde unterschieden zwischen keinen, schwachen und starken Effekten. Es wurden alle Kombinationen von Effekten simuliert und iterativ 100 mal durchgeführt.

Die Simulation wurde für jede Kombination aus keinen, schwachen und starken Effekten sowohl für die Haupt- als auch die Interaktionseffekte durchgeführt. Durch die hohe Laufzeit, die bereits beim Anwendungsfall bei der Methode GGLasso festgestellt wurde, wurde die Anzahl an Iterationen pro Kombination auf 100 beschränkt.

Der lineare Prädiktor $Y_{lin}$ wurde anschließend mit:

$$Y_{WK} := \frac{1}{1 + \exp(-Y_{lin})}$$

reskaliert zu Wahrscheinlichkeiten. Mit Hilfe der Wahrscheinlichkeiten $Y_{WK}$ wurde die binäre Zielgröße $Y$ durch eine Bernoulli-Verteilung simuliert.

Neben den Einflussvariablen wurden noch 100 $\mathcal{N}(0,1)$ standardnormalverteilte Störvariablen $S_1, \ldots, S_{100}$ simuliert. Es wurden pro Iteration 1.000 Beobachtungen generiert. Für alle Methoden wurden logistische Regressionsmodelle geschätzt in der Form:

$$\begin{aligned} Y \sim \quad & \beta_1 X_1 + \beta_2 X_2 + \theta_1 X_1 X_2 + \\ & \beta_{3A} X_{3A} + \beta_{3B} X_{3B} + \beta_{3C} X_{3C} + \beta_{3D} X_{3D} + \beta_{3E} X_{3E} + \\ & \theta_{2A} X_1 X_{3A} + \theta_{2B} X_1 X_{3B} + \theta_{2C} X_1 X_{3C} + \theta_{2D} X_1 X_{3D} + \theta_{2E} X_1 X_{3E} + \\ & \sum_{i=1}^{100} \beta_{i+3} S_i + \sum_{i=1}^{100} \theta_{i+2} X_1 S_i + \varepsilon \end{aligned}$$

Bei GLINTERNET und GGLasso wurden die Dummy Variablen von $X_3$ bzw. $X_1 X_3$ zusammen gruppiert. Für alle Verfahren wurde eine Folge von $k = 50$ Tuning-Parametern

$\lambda_k$ festgelegt. Bei allen Iterationen und Verfahren wurde anschließend festgehalten bei welchem Parameter $\lambda_k$ die Haupt- und Interaktionsterme das erste Mal in das Modell aufgenommen wurden. Anschließend wurde daraus eine Rangliste abgeleitet. Mit Hilfe der Rangliste wurde untersucht, welchen Platz in der Rangliste die Einflussvariablen $X_1$, $X_2$ und $X_3$ und die Interaktionen $X_1X_2$ und $X_1X_3$ hatten. Es wurden dabei, wie im Anwendungsfall in Kapitel 4, die Haupt- und Interaktionseffekte getrennt betrachtet.

Bei den Verfahren Pliable und GAMSEL wurde jeweils das minimale $\lambda_k$ der Ausprägungen von $X_3$ gewählt. Für beide Verfahren ist es damit einfacher die kategorielle Variable und ihre Interaktionen in das Modell zu selektieren, da lediglich eine Ausprägungen hinzugefügt werden muss.

### 5.2.2 Ergebnisse

Bei der Evaluation wurde die Performance der vier Modelle festgestellt und miteinander verglichen. Dafür wurde für die verschiedenen Kombinationen aus Effekten für Haupt- und Interaktionsterme der Rang untersucht. Es wurde analysiert bei wie vielen der 100 Iterationen die Haupt- bzw. Interaktionsterme innerhalb der ersten $N$ Ränge waren. Die Ranglisten wurden dabei, wie beim Anwendungsfall, getrennt betrachtet für die Haupt- und Interaktionseffekte. Für die drei Haupteffekte wurde untersucht wie häufig diese in den ersten drei bis zehn Rängen vorkamen. Bei den zwei Interaktionseffekten wurde der Anteil für zwei bis zehn ermittelt. Die Evaluation der verschiedenen Kombinationen aus Effekten für Haupt- und Interaktionsterme erfolgte mittels Grafiken auf den nächsten Seiten. Ein Lesebeispiel für die starken Haupteffekte und starken Interaktionen in Abbildung 5.3 verdeutlicht die Idee der Evaluation:

Die kategorielle Variable $X_3$ war bei dem Verfahren GGLasso bei ungefähr 30% der Iterationen innerhalb der ersten 3 Ränge zu finden, bei 50% der Iterationen innerhalb der ersten 4 Ränge und 65% innerhalb der ersten 5 Ränge. Beim Verfahren GLINTERNET war die Variable $X_3$ bei 100% der Stichproben innerhalb der ersten 3 Ränge. Damit schnitt GLINTERNET bzgl. der Selektion der Variable X3 besser ab als GGLasso.

Im Idealfall selektierte ein Verfahren alle Einflussvariablen innerhalb der ersten drei, bzw. bei Interaktionen innerhalb der ersten zwei Ränge. Bei der Evaluation der Grafiken ist zu beachten, dass in Abbildung 5.4 in Reihe eins und zwei die Haupteffekte falsch positiv und in Reihe drei und vier die Interaktionseffekte falsch positiv sind.

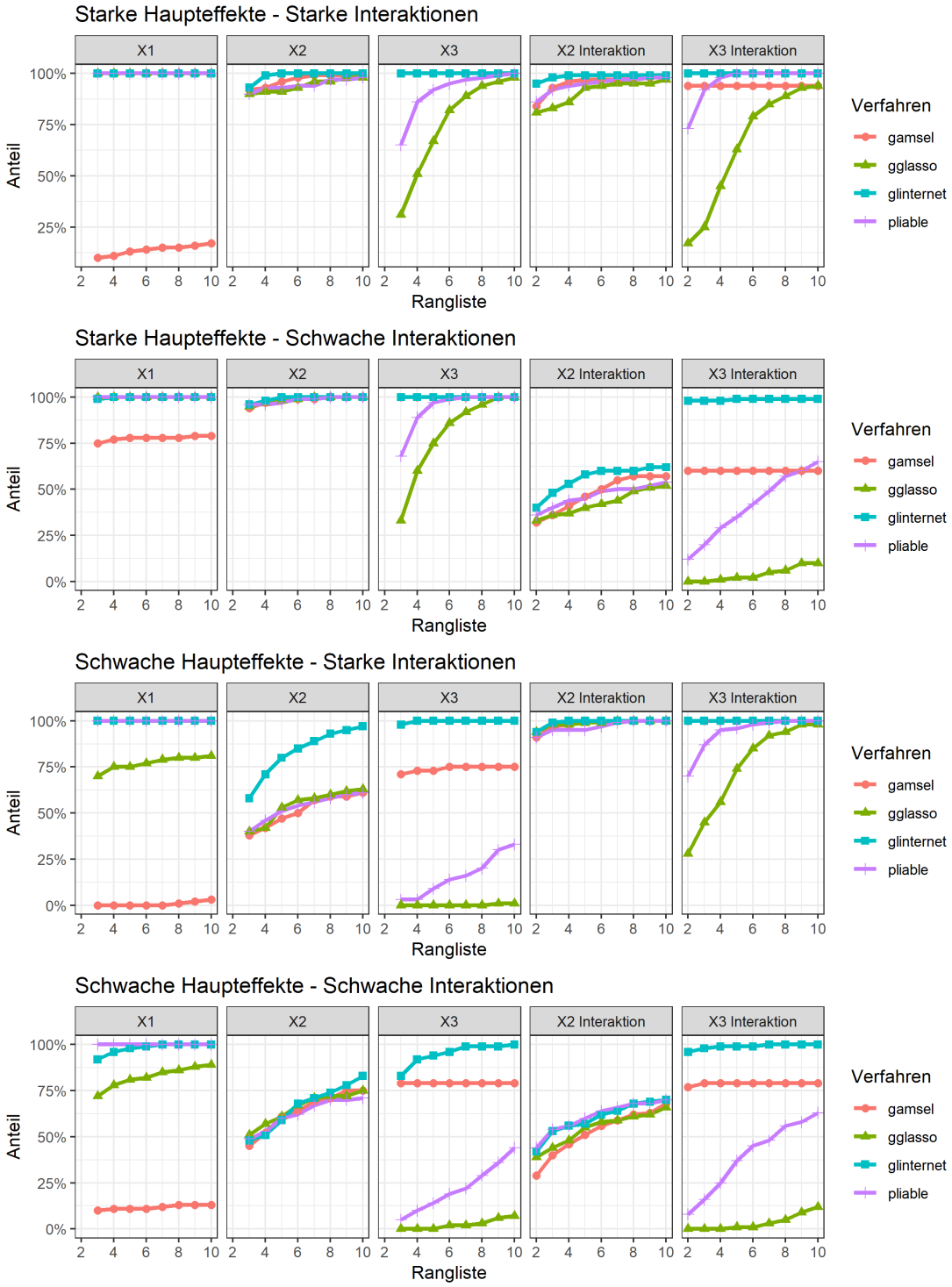


**Abb. 5.3:** Der erste Teil der Simulationsstudie:
$\beta_1 = 0{,}4$, $\beta_2 = 0{,}4$, $\beta_3 = 1$, $\theta_1 = 0{,}4$ und $\theta_2 = 1$ (1. Reihe)
$\beta_1 = 0{,}4$, $\beta_2 = 0{,}4$, $\beta_3 = 1$, $\theta_1 = 0{,}2$ und $\theta_2 = 0{,}5$ (2. Reihe)
$\beta_1 = 0{,}2$, $\beta_2 = 0{,}2$, $\beta_3 = 0{,}5$, $\theta_1 = 0{,}4$ und $\theta_2 = 1$ (3. Reihe)
$\beta_1 = 0{,}2$, $\beta_2 = 0{,}2$, $\beta_3 = 0{,}5$, $\theta_1 = 0{,}2$ und $\theta_2 = 0{,}5$ (4. Reihe)

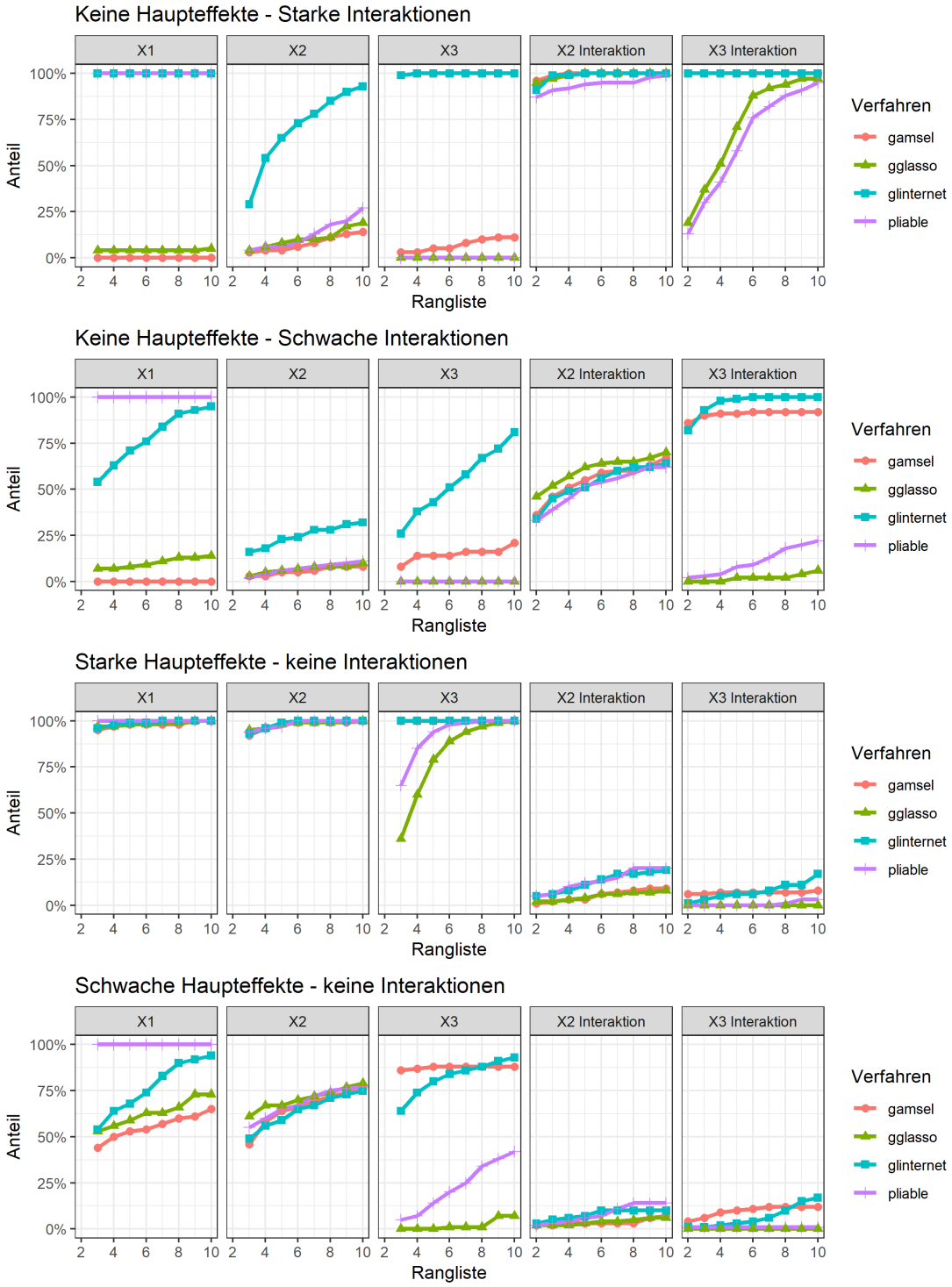


**Abb. 5.4:** Der zweite Teil der Simulationsstudie:
$\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0$, $\theta_1 = 0{,}4$ und $\theta_2 = 1$ (1. Reihe)
$\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0$, $\theta_1 = 0{,}2$ und $\theta_2 = 0{,}5$ (2. Reihe)
$\beta_1 = 0{,}4$, $\beta_2 = 0{,}4$, $\beta_3 = 1$, $\theta_1 = 0$ und $\theta_2 = 0$ (3. Reihe)
$\beta_1 = 0{,}2$, $\beta_2 = 0{,}2$, $\beta_3 = 0{,}5$, $\theta_1 = 0$ und $\theta_2 = 0$ (4. Reihe)

Die Methode GAMSEL zeigte vor allem bei starken Interaktionen ein großes Problem die Variable $X_1$ zu identifizieren (siehe Abbildung 5.3, 1. und 3. Reihe). Im Fall ohne Interaktionen zeigte sich bei GAMSEL ein ähnliches Bild wie bei den restlichen Verfahren (siehe Abbildung 5.4, 3. und 4. Reihe). Das Problem lag nicht bei der Variable $X_2$ vor. Eine mögliche Erklärung wäre, dass die Interaktion von $X_1X_3$ den Haupteffekt von $X_1$ überlagerte. Es ist zudem auffallend, dass bei der Interaktion $X_1X_3$ die Linie bei allen Durchläufen fast horizontal verläuft. Die Interaktion wurde entweder als erstes oder erst bei einem hohen $k$ bzw. überhaupt nicht in das Modell aufgenommen. GAMSEL zeigte somit sowohl im Anwendungsfall als auch in der Simulationsstudie hohe Varianz.

GGLasso hatte große Schwächen bei der Identifikation der kategoriellen Variable $X_3$ und der Interaktion $X_1X_3$. Die Gruppierung der Dummy Variablen der kategoriellen Variable $X_3$ und der Interaktion $X_1X_3$ führte nicht zu einer effektiven Selektion der Einflussvariablen. Bei den stetigen Variablen und der Interaktion $X_1X_2$ schnitt GGLasso durchschnittlich ab.

GLINTERNET zeigt in der Abbildung 5.3 eine gute Performance. Alle Variablen und Interaktionen wurden hier effektiv identifiziert. Bei den reinen Interaktionseffekten selektierte GLINTERNET zwangsweise fälschlicherweise die Haupteffekte, um die starke Hierarchiebedingung zu erfüllen. Trotz dieser Restriktion wurden auch die reinen Interaktionen früh in die Modelle aufgenommen. Bei den Durchläufen ohne Interaktionseffekte zeigte sich eine leicht erhöhte Falsch-positiv-Rate bei der Interaktion von $X_1X_3$.

Bei Pliable ist zu berücksichtigen, dass die Variable $X_1$ nicht bestraft wurde und somit in allen Iterationen immer im Modell enthalten war. Dies war zwar in den meisten Fällen kein Problem, führte jedoch bei den Durchläufen ohne Haupteffekte zu einer 100%-igen Falsch-positiv-Rate. Pliable hatte ebenfalls Probleme mit der kategoriellen Variable $X_3$. Trotz der schwachen Bedingung, dass nur eine Ausprägung ins Modell aufgenommen werden musste, selektierte Pliable insbesondere den Haupteffekt erst bei einem niedrigem Strafterm. Bei der Interaktion mit der stetigen Variable $X_2$ zeigt die Methode eine ähnliche Performance wie die anderen Verfahren.

Insgesamt überzeugte die Methode GLINTERNET bei allen Durchläufen mit guten Ergebnissen. GGLasso und Pliable zeigten beide erhebliche Probleme mit der kategoriellen Variable und der Interaktion mit dieser. Die Annahme aus dem Anwendungsfall, dass die beiden Verfahren die Relevanz von kategoriellen Variablen unterschätzen, bestätigt sich hier. Bei GAMSEL lag hohe Varianz in den Modellen vor und insgesamt war die Methode ungeeignet für die Modellierung von Interaktionen.

# 6 Fazit

Abschließend folgt eine Zusammenfassung der gewonnen Erkenntnisse in dieser Thesis und ein Ausblick welche weiteren Fragestellungen sich ergeben haben.

## 6.1 Fazit

Ziel dieser Arbeit war es Haupt- und Interaktionsterme zu einer ausgewählten Variable in hochdimensionalen Datensätzen mit regularisierten, additiven Regressionsmodellen zu identifizieren. Dazu wurden vier geeignete Methoden recherchiert und in Kapitel 3 wurden deren mathematische Grundlagen miteinander verglichen. Die vier Methoden GGLasso, GLINTERNET, Pliable und GAMSEL basieren zwar alle auf der (logistischen) Regression mit Lasso Regularisierung, unterscheiden sich jedoch in vielen Aspekten. Bei den Unterschieden sind insbesondere der Umgang mit Interaktionen in Form von hierarchischen Bedingungen, die Möglichkeit in Gruppen zu regularisieren und die Modellierung nicht-linearer Effekte hervorzuheben. Die Verfahren Pliable und GLINTERNET sind speziell für die Modellierung und Regularisierung von Interaktionen entwickelt.

In einem praktischen Anwendungsfall wurden die vier Methoden auf einem Datensatz von Nierentransplantationen angewandt. Aus historischen Daten von 2015 bis 2019 wurde die Wahrscheinlichkeit für Versterben oder Nierenversagen innerhalb eines Jahres geschätzt. Dabei wurden sowohl Haupt- als auch Interaktionsterme zwischen der Nierenqualität (KDPI) und medizinischen Merkmalen des Empfängers untersucht. Hierbei war es das Ziel mittels der Verfahren relevante Variablen zu identifizieren, um ein besseres Verständnis für Einflussfaktoren bei Nierentransplantationen entwickeln zu können. Zudem sollten die Erkenntnisse in einem Dashboard für Anwender ohne statistisches Fachwissen verständlich verfügbar gemacht werden.

Nach umfangreicher Datenaufbereitung konnten die Modelle trainiert werden und es wurden Ranglisten für die Relevanz für Variablen und Interaktionen abgeleitet. Auf Basis der Ranglisten wurden grafisch verschiedene Teilmengen an Haupt- und Interaktionseffekten festgelegt. Mit den Teilmengen wurden logistische Regressionsmodelle trainiert und ein bestes Modell mittels Kreuzvalidierung bestimmt. Die Unterschiede waren trotz der unterschiedlichen Teilmengen häufig nur marginal. Insgesamt verbesserten die Modelle mit Interaktionen den Kreuzvalidierungsfehler nur minimal im Vergleich zu Modellen ohne Interaktionen. Mittels der Verfahren konnten somit Haupt- und Interaktionsterme identifiziert werden, die die Prognose der Überlebenswahrscheinlichkeit beeinflussen. Es ist festzuhalten, dass die Variablenselektion ein

kontinuierlicher Prozess ist und eine beste Teilmenge nicht mit einer einfachen Metrik bestimmt werden kann.

In einer R-Shiny App wurden die Erkenntnisse der Daten und des Modells, sowie die relevanten Interaktionseffekte mit Grafiken und Metriken dargestellt. Der Einfluss der Variablen auf die Prognose und speziell der Effekt der Interaktionen konnte in einem Dashboard vermittelt werden. Das Dashboard war mit Hilfe von Erklärungstexten auch ohne statisches Fachwissen verständlich.

Um die Methoden systematisch miteinander zu vergleichen, wurden die Ergebnisse aus dem Anwendungsfall untersucht und eine Simulationsstudie wurde durchgeführt. In der Simulationsstudie wurde die Performance der Verfahren geprüft mit verschiedenen Kombinationen aus Effekten für Haupt-und Interaktionsterme. Bei den Verfahren Pliable und GGLasso konnten große Probleme bei der Identifikation von kategoriellen Variablen und stetigen Variablen mit wenigen Ausprägungen festgestellt werden. Die Methode GAMSEL zeigte sowohl im Anwendungsfall als auch in der Simulationsstudie hohe Varianz zwischen den Modellen und war insgesamt ungeeignet zur Modellierung von Interaktionen. Das Verfahren GLINTERNET konnte in vielen Szenarien die anderen Modelle deutlich übertreffen und zeigte sich im speziellen Fall mit Interaktionen zu einer ausgewählten Variable als die beste Wahl.

Die gute Erklärbarkeit machen additive Regressionsmodelle besonders interessant, insbesondere wenn die Modelle nur unterstützend genutzt werden. Durch zahlreiche Erweiterungen können Regressionsmodelle auch an komplexe Anwendungsfälle angepasst werden. Hierbei ist es essentiell eine passende Methode vorab zu recherchieren.

## 6.2 Ausblick

Da die Kreuzvalidierung in einigen der vier Verfahren nicht funktionierte, wurde im praktischen Anwendungsfall mit Nierentransplantationen ein Verfahren mit Bootstrapping entwickelt um eine Rangliste für Variablen und Interaktionen zu entwickeln. In einer Fortsetzung könnte den Fehlern auf den Grund gegangen werden um anschließend die Methoden mit der üblichen Kreuzvalidierungsmethode durchzuführen.

Der Fokus in dieser Thesis lag auf der Variablenselektion mit Interaktionen mit Hilfe der vier speziellen Regressionsmodelle. Ein Vergleich mit nichtparametrischen Methoden, wie beispielsweise Random Forests, wäre eine denkbare Erweiterung der Analyse. Hierbei wäre es interessant, ob sich ein Kompromiss zwischen Erklärbarkeit und

Performance im Kontext von Identifikation von Interaktionen feststellen lässt. Zudem könnte auch ein Feintuning der Parameter beim Training der Modelle vorgenommen werden.

Im praktischen Anwendungsteil wurde die Adjustierung mittels Residuen zum Umgang mit Confounding benutzt. Um die Unterschiede der Adjustierung und die Probleme des Confoundings zu untersuchen, wäre es eine interessante Fortsetzung weitere Durchläufe der Methoden ohne Adjustierung zu untersuchen. Zudem könnten in diesem Kontext noch weitere Verfahren zum Umgang mit Confounding entwickelt und getestet werden.

# Quellenverzeichnis

*A Guide to Calculating and Interpreting the Estimated Post-Transplant Survival* (Apr. 2020). https://optn.transplant.hrsa.gov/media/1511/guide_to_calculating_interpreting_epts.pdf. Accessed: 2021-03-11.

Austin, P. C. (2011). „An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies". In: *Multivariate Behavioral Research* 46.3. PMID: 21818162, S. 399–424. DOI: 10.1080/00273171.2011.568786. eprint: https://doi.org/10.1080/00273171.2011.568786.

– (2018). „Assessing the performance of the generalized propensity score for estimating the effect of quantitative or continuous exposures on binary outcomes". In: *Statistics in Medicine* 37.11, S. 1874–1894. DOI: https://doi.org/10.1002/sim.7615. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.7615. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.7615.

Bien, J.; J. Taylor & R. Tibshirani (Juni 2013). „A lasso for hierarchical interactions". In: *The Annals of Statistics* 41.3, S. 1111–1141. ISSN: 0090-5364. DOI: 10.1214/13-aos1096. URL: http://dx.doi.org/10.1214/13-AOS1096.

Chopra, B. & K. K. Sureshkumar (Juni 2015). „Changing organ allocation policy for kidney transplantation in the United States". In: DOI: 10.5500/wjt.v5.i2.38. URL: https://www.wjgnet.com/2220-3230/full/v5/i2/38.htm.

Chouldechova, A. & T. Hastie (2015). *Generalized Additive Model Selection.* arXiv: 1506.03850 [stat.ML].

Cox, D. R. (1984). „Interaction". In: *International Statistical Review / Revue Internationale de Statistique* 52.1, S. 1–24. ISSN: 03067734, 17515823.

Demmler, A. & C. Reinsch (1975). „Oscillation Matrices with Spline Smoothing." In: *Numerische Mathematik* 24, S. 375–382. URL: http://eudml.org/doc/132350.

*Die Nierentransplantation* (o.D.). https://www.organspende-info.de/organspende/transplantierbare-organe/nierentransplantation.html. Accessed: 2020-01-14.

Dinga, R.; L. Schmaal; B. Penninx; D. Veltman & A. Marquand (Aug. 2020). „Controlling for effects of confounding variables on machine learning predictions". In: DOI: 10.1101/2020.08.17.255034.

*Estimated Glomerular Filtration Rate (eGFR)* (Sep. 2020). https://www.kidney.org/atoz/content/gfr. Accessed: 2021-04-28.

Hastie, T.; R. Tibshirani & J. Friedman (2009). *The elements of statistical learning: data mining, inference and prediction.* 2. Aufl. Springer. URL: http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

Hirano, K. & G. W. Imbens (2004). „The Propensity Score with Continuous Treatments". In: *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives.* John Wiley & Sons, Ltd. Kap. 7, S. 73–84. ISBN: 9780470090459. DOI: https://doi.org/10.1002/0470090456.ch7. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470090456.ch7. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470090456.ch7.

Höfler, M. (Sep. 2005). „Causal inference based on counterfactuals". In: *BMC Medical Research Methodology* 5. DOI: 10.1186/1471-2288-5-28.

James, G.; D. Witten; T. Hastie & R. Tibshirani (2013). *An Introduction to Statistical Learning: with Applications in R.* Springer.

Kabore, R.; M. C. Haller; J. Harambat; G. Heinze & K. Leffondré (Feb. 2017). „Risk prediction models for graft failure in kidney transplantation: A systematic review". In: *Nephrology, dialysis, transplantation : official publication of the European Dialysis and Transplant Association - European Renal Association* 32. DOI: 10.1093/ndt/gfw405.

Lim, M. & T. Hastie (2013). *Learning interactions through hierarchical group-lasso regularization.* arXiv: 1308.2719 [stat.ME].

Mai, M. (2016). *Nierenfunktion.* URL: https://www.netdoktor.de/anatomie/nierenfunktion/.

Meier, L.; S. Van De Geer & P. Bühlmann (2008). „The group lasso for logistic regression". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1, S. 53–71. DOI: https://doi.org/10.1111/j.1467-9868.2007.00627.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9868.2007.00627.x. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2007.00627.x.

Mohan, S.; M. C. Chiles; R. E. Patzer; S. Pastan; S. A. Husain; D. Carpenter; G. K. Dube; R. J. Crew; L. E. Ratner & D. J. Cohen (Mai 2018). „Factors leading to the discard of deceased donor kidneys in the United States". In: DOI: 10.1016/j.kint.2018.02.016.

Oberkofler, C. & O. de Rougemont (2015). „Pankreastransplantation". In: *Der Diabetologe*, S. 538–544. DOI: 10.1007/s11428-015-0019-8. URL: https://link.springer.com/article/10.1007%5C%2Fs11428-015-0019-8.

*Organ Procurement and Transplantation Network* (o.D.). https://optn.transplant.hrsa.gov/data/view-data-reports/national-data/. Accessed: 2020-01-14.

*Policies* (o.D.). https://optn.transplant.hrsa.gov/governance/policies/. Accessed: 2020-01-14.

Robins, J.; M. Hernán & B. Brumback (2000). „Marginal structural models and causal inference in epidemiology“. In: *Epidemiology (Cambridge, Mass.)* 11.5, S. 550–560. ISSN: 1044-3983. DOI: 10.1097/00001648-200009000-00011. URL: https://doi.org/10.1097/00001648-200009000-00011.

Rosenbaum, P. R. & D. B. Rubin (Apr. 1983). „The central role of the propensity score in observational studies for causal effects“. In: *Biometrika* 70.1, S. 41–55. ISSN: 0006-3444. DOI: 10.1093/biomet/70.1.41. eprint: https://academic.oup.com/biomet/article-pdf/70/1/41/662954/70-1-41.pdf. URL: https://doi.org/10.1093/biomet/70.1.41.

Roulston, M. S. (2007). „Performance targets and the Brier score“. In: *Meteorological Applications* 14.2, S. 185–194. DOI: https://doi.org/10.1002/met.21. eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/met.21. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/met.21.

Rudin, C. (2019). *Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead.* arXiv: 1811.10154 [stat.ML].

Schönemann, C. (Feb. 2016). „HLA-Antikörper vor und nach Nierentransplantation“. In: URL: https://www.trillium.de/zeitschriften/trillium-diagnostik/ausgaben-2016/td-22016/schwerpunkt/hla-antikoerper-vor-und-nach-nierentransplantation.html.

Sengers, H.; J. Houtzager; M. Idu; M. Heemskerk; E. Heurn; J. Heide; J. Kers; S. Berger; T. Gulik & F. Bemelman (Apr. 2019). „Impact of Cold Ischemia Time on Outcomes of Deceased Donor Kidney Transplantation: An Analysis of a National Registry“. In: *Transplantation Direct* 5, S. 1. DOI: 10.1097/TXD.0000000000000888.

*The New Kidney Allocation System* (o.D.). https://optn.transplant.hrsa.gov/media/1235/kas_faqs.pdf. Accessed: 2020-01-14.

Tibshirani, R. (1996). „Regression Shrinkage and Selection via the Lasso“. In: *Journal of the Royal Statistical Society (Series B)* 58, S. 267–288.

Tibshirani, R. & J. Friedman (2018). *A Pliable Lasso.* arXiv: 1712.00484 [stat.ME].

Wirth, R. (2000). „CRISP-DM: Towards a standard process model for data mining“. In: *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, S. 29–39.

Wright, M.; A. Ziegler & I. König (März 2016). „Do little interactions get lost in dark random forests?“ In: *BMC Bioinformatics* 17, S. 145. DOI: 10.1186/s12859-016-0995-8.

Yan, X. & J. Bien (2017). „Hierarchical Sparse Modeling: A Choice of Two Group Lasso Formulations“. In: *Statistical Science* 32.4, S. 531–560. ISSN: 0883-4237. DOI: 10.1214/17-sts622. URL: http://dx.doi.org/10.1214/17-STS622.

Yang, Y. & H. Zou (Aug. 2014). „A fast unified algorithm for solving group-lasso penalize learning problems“. In: *Statistics and Computing* 25. DOI: 10.1007/s11222-014-9498-5.

Yuan, M. & Y. Lin (2006). „Model selection and estimation in regression with grouped variables“. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1, S. 49–67. DOI: https://doi.org/10.1111/j.1467-9868.2005.00532.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9868.2005.00532.x. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2005.00532.x.

Zschiedrich, S.; A. Kramer-Zucker; B. Jänigen; M. Seidl; F. Emmerich; P. Pisarski & T. B. Huber (2015). „An update on ABO-incompatible kidney transplantation“. In: *Transplant International* 28.4, S. 387–397. DOI: https://doi.org/10.1111/tri.12485. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/tri.12485. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/tri.12485.

# Anhang

## A Pakete

**Code A.1:** Die genutzten R-Pakete und Versionen.

```
> packageVersion("base")
[1] 4.0.3
> packageVersion("haven")
[1] 2.3.1
> packageVersion("dplyr")
[1] 1.0.3
> packageVersion("plyr")
[1] 1.8.6
> packageVersion("tidyverse")
[1] 1.3.0
> packageVersion("data.table")
[1] 1.13.6
> packageVersion("boot")
[1] 1.3.25
> packageVersion("scales")
[1] 1.1.1
> packageVersion("ggplot2")
[1] 3.3.3
> packageVersion("gridExtra")
[1] 2.3
> packageVersion("grid")
[1] 4.0.3
> packageVersion("effects")
[1] 4.2.0
> packageVersion("scales")
[1] 1.1.1
> packageVersion("pliable")
[1] 1.1.1
> packageVersion("gamsel")
[1] 1.8.1
> packageVersion("glinternet")
[1] 1.0.11
> packageVersion("gglasso")
[1] 1.5
> packageVersion("splines")
[1] 4.0.3
> packageVersion("shiny")
[1] 1.6.0
> packageVersion("shinydashboard")
[1] 0.7.1
```

# B Programmcode

**Code B.1:** Bootstrap GGLasso

```
library(gglasso)
library(dplyr)
library(plyr)


###Datensatz vorbereiten

source("functions/var_selection.R")
set.seed(123)
gglasso_df = data.frame()
n = 50

###Funktion um Gruppen zu erstellen

get_numlevels = function(x){
  if (is.factor(x)){
    length(unique(x))-1
  }
  else 1
}

###Funktion um erstes Auftreten der Variable zu bestimmen

min_level = function(x){
  min(which(x!=0))
}

###Gruppierung festlegen

X = model.matrix(~ . + KDPI * . , dplyr::select(complete_cases, -Status))[, -1]
numLevels = as.vector(sapply(complete_cases %>% select(-Status),get_numlevels))
groups = c()
for (i in 1:length(numLevels)){
  groups = c(groups, rep(i, numLevels[i]))
}
groups = c(groups, groups+max(groups))
KDPI = which(colnames(X) == "KDPI")+ sum(numLevels)
groups = groups[-(KDPI)]
groups[KDPI:length(groups)] = groups[KDPI:length(groups)]-1

###Strafterm anpassen um KDPI nicht zu regularisieren

penalty = sqrt(as.vector(table(groups)))
penalty[which(names(complete_cases) == "KDPI")] = 0

###Gruppen kombinieren (zur Evaluation)

combine_groups = c()

for (i in 1:length(numLevels)){
  combine_groups = c(combine_groups, rep(names(complete_cases[i]), numLevels[i]))
}

```

```
combine_groups = c(combine_groups, paste0(combine_groups, ":KDPI"))
combine_groups = combine_groups[-which(combine_groups == "KDPI:KDPI")]

###Iterationen

for (i in 1:n){
  print(paste0("Progress: ", round((i-1)*100/n), "%"))

  ###Resample

  resample = complete_cases[sample(dim(complete_cases)[1], replace = T), ]

  ####GGLasso trainieren

  X = model.matrix(~ . + KDPI * . , dplyr::select(resample, -Status))[, -1]
  Y = as.integer(resample$Status)-1
  Y[Y==0] = -1
  group_lasso_fit = gglasso(x = X, y = Y, group = groups, nlambda = 50, lambda.factor
 ↪= 0.00005, pf = penalty, loss = "logit")

  ###Erstes Auftreten jeder Variable bestimmen

  beta = t(group_lasso_fit$beta)
  min_lambda = apply(beta, 2,min_level)
  min_lambda[is.infinite(min_lambda)] = NA


  ###Dataframe erstellen

  temp_df = as.data.frame(t(min_lambda))
  colnames(temp_df) = combine_groups
  temp_df = temp_df[, !duplicated(colnames(temp_df))]
  gglasso_df = rbind.fill(gglasso_df, temp_df)
}

###Dataframe speichern

save(gglasso_df ,file="data/Processed Data/gglasso.Rda")
```

**Code B.2:** Bootstrap GLINTERNET

```
library(dplyr)
library(plyr)
library(glinternet)

###Datensatz vorbereiten

source("functions/var_selection.R")
set.seed(123)
glinternet_df = data.frame()
n = 50

###Funktion um kategoriellen Variablen in aufeinanderfolgende Zahlen umzukodieren

code_factors = function(x){
  if (is.factor(x)){
    as.numeric(factor(x))-1
```

```
17    }
18    else  x
19  }
20
21  ###Funktion um die Anzahl Ausprägungen pro Kategorie zu bestimmen
22
23  get_numlevels = function(x){
24    if (is.factor(x)){
25      length(unique(x))
26    }
27    else 1
28  }
29
30  ###Iterationen
31
32  for (i in 1:n){
33    print(paste0("Progress: ", round((i-1)*100/n), "%"))
34
35    ###Resample
36
37    resample = complete_cases[sample(dim(complete_cases)[1], replace = T), ]
38
39    ###GLINTERINET trainieren
40
41    X = sapply(resample %>% select(-Status), code_factors)
42    Y = as.integer(resample$Status)-1
43    numLevels = as.vector(sapply(complete_cases %>% select(-Status),get_numlevels))
44    glinternet_fit = glinternet(X=X, Y=Y, family = "binomial", numLevels=numLevels,
    ↪interactionCandidates=21)
45
46    ###Koeffizienten festlegen
47
48    coefs = coef(glinternet_fit)
49    maineffect_cat = c()
50    maineffect_num = c()
51    int_cat = c()
52    int_num = c()
53    for (i in 1:50){
54      maineffect_cat = c(maineffect_cat, coefs[[i]]$mainEffects$cat)
55      maineffect_num = c(maineffect_num, coefs[[i]]$mainEffects$cont)
56      int_cat = c(int_cat, coefs[[i]]$interactions$catcont[,1])
57      temp_num = c(coefs[[i]]$interactions$contcont[,1], coefs[[i]]$interactions$
    ↪contcont[,2])
58      temp_num = temp_num[temp_num != 9]
59      int_num = c(int_num, temp_num)
60    }
61
62    ###Namen der Variablen bestimmen
63
64    factor_names = names(complete_cases %>% select(-Status))[sapply(complete_cases %>%
    ↪select(-Status), is.factor)]
65    num_names = names(complete_cases)[sapply(complete_cases, is.numeric)]
66
67    ###Erstes Auftreten jeder kategoriellen Variable
68
69    first_main_effect_cat = 50-table(maineffect_cat)
70    first_main_effect_num = 50-table(maineffect_num)
71    first_int_effect_cat = 50-table(int_cat)
```

```
  first_int_effect_num = 50-table(int_num)

  ####Namen anhängen

  main_effect_cat_names = c()
  main_effect_num_names = c()
  int_effect_cat_names = c()
  int_effect_num_names = c()

  for (i in names(first_main_effect_cat)){
    i = as.numeric(i)
    main_effect_cat_names = c(main_effect_cat_names, factor_names[i])
  }
  for (i in names(first_main_effect_num)){
    i = as.numeric(i)
    main_effect_num_names = c(main_effect_num_names, num_names[i])
  }
  for (i in names(first_int_effect_cat)){
    i = as.numeric(i)
    int_effect_cat_names = c(int_effect_cat_names, factor_names[i])
  }
  for (i in names(first_int_effect_num)){
    i = as.numeric(i)
    int_effect_num_names = c(int_effect_num_names, num_names[i])
  }

  names(first_main_effect_cat) = main_effect_cat_names
  names(first_main_effect_num) = main_effect_num_names
  names(first_int_effect_cat) = paste0(int_effect_cat_names, ":KDPI")
  names(first_int_effect_num) =  paste0(int_effect_num_names, ":KDPI")


  ###Dataframe erstellen

  first_main_effect_cat = data.frame(rbind(first_main_effect_cat))
  first_main_effect_num = data.frame(rbind(first_main_effect_num))
  first_int_effect_cat = data.frame(rbind(first_int_effect_cat))
  first_int_effect_num = data.frame(rbind(first_int_effect_num))

  temp_df = cbind(first_main_effect_cat, first_main_effect_num, first_int_effect_cat,
  ↪first_int_effect_num)
  rownames(temp_df) = NULL

  glinternet_df = rbind.fill(glinternet_df, temp_df)
}

###Dataframe speichern

save(glinternet_df ,file="data/Processed Data/glinternet.Rda")
```

**Code B.3:** Bootstrap Pliable

```
library(pliable)
library(dplyr)
library(plyr)


```

```
6  ###Datensatz vorbereiten
7
8  source("functions/var_selection.R")
9  set.seed(123)
10 pliable_df = data.frame()
11 n = 50
12
13
14 ###Funktion um erstes Auftreten der Variable zu bestimmen
15
16 min_level = function(x){
17   min(which(x!=0))
18 }
19
20 ###Iterationen
21
22 for (i in 1:n){
23   print(paste0("Progress: ", round((i-1)*100/n), "%"))
24
25   ###Resample
26
27   resample = complete_cases[sample(dim(complete_cases)[1], replace = T), ]
28
29   ###Pliable trainieren
30
31   X = model.matrix(~ . , dplyr::select(resample, -c(Status,KDPI)))[, -1]
32   Y = as.integer(resample$Status)-1
33   pliable_fit = pliable(X, z = resample$KDPI, y = Y,
34                         family = "binomial",
35                         lambda.min.ratio = 0.000005,
36                         thr = 0.1,
37                         tt = 0.1^7)
38
39
40   ###Dataframe für Haupteffekt bestimmen
41
42   beta = t(pliable_fit$beta)
43   min_lambda = apply(beta, 2,min_level)
44   min_lambda[is.infinite(min_lambda)] = NA
45   main_df = as.data.frame(t(min_lambda))
46   colnames(main_df) = colnames(X)
47   betaz = data.frame(KDPI = 1)
48
49   ###Dataframe für Interaktionen bestimmen
50
51   theta = t(pliable_fit$theta[,1,])
52   min_lambda = apply(theta, 2,min_level)
53   min_lambda[is.infinite(min_lambda)] = NA
54   int_df = as.data.frame(t(min_lambda))
55   colnames(int_df) = paste0(colnames(X), ":KDPI")
56
57   ###Dataframes zusammenführen
58
59   temp_df = cbind(main_df, betaz, int_df)
60   pliable_df = rbind.fill(pliable_df, temp_df)
61 }
62
63 ###Dataframe abspeichern
```

```

save(pliable_df ,file="data/Processed Data/pliable.Rda")
```

**Code B.4:** Bootstrap GAMSEL

```
library(gamsel)
library(dplyr)
library(plyr)

###Datensatz vorbereiten

source("functions/var_selection.R")
set.seed(123)
gamsel_lin_df = data.frame()
gamsel_non_lin_df = data.frame()
n = 50

###Funktion um Dummy Variablen nach Training wieder zusammenzuführen

get_numlevels = function(x){
  if (is.factor(x)){
    length(unique(x))-1
  }
  else 1
}

###Funktion um den degree festzulegen für GAMSEL

set_degree = function(x){
  if (length(unique(x))<10){
    1
  }
  else 10
}

###Funktion um erstes Auftreten der Variable zu bestimmen

min_level = function(x){
  min(which(x!=0))
}

##Degree und Anzahl Level pro Variable bestimmen

numLevels = as.vector(sapply(complete_cases %>% select(-Status),get_numlevels))
degree_level = as.vector(sapply(complete_cases %>% select(-Status),set_degree))
degrees = c()
for (i in 1:length(numLevels)){
  if (numLevels[i] == 1){
    degrees = c(degrees, degree_level[i])
  }
  else degrees = c(degrees, rep(1, numLevels[i]))
}
degrees = c(degrees, degrees)

###Namen bestimmen

beta_names = c()
```

```
for (i in 1:dim(X)[2]){
  beta_names = c(beta_names, rep(colnames(X)[i], degrees[i]))
}

###Iterationen

for (i in 1:n){
  print(paste0("Progress: ", round((i-1)*100/n), "%"))

  ###Resample

  resample = complete_cases[sample(dim(complete_cases)[1], replace = T), ]

  ###GAMSEL trainieren

  X = model.matrix(~ . + KDPI * . , dplyr::select(resample, -Status))[, -1]
  Y = as.integer(resample$Status)-1
  gamsel_fit = gamsel(X,Y, family = "binomial", traceit = F, degrees = degrees)

  ###Erstes Auftreten jeder Variable bestimmen (linear)

  alpha = t(gamsel_fit$alphas)
  min_lambda = apply(alpha, 2,min_level)
  min_lambda[is.infinite(min_lambda)] = NA

  ###Dataframe erstellen (linear)

  temp_df = as.data.frame(t(min_lambda))
  colnames(temp_df) = colnames(X)
  gamsel_lin_df = rbind.fill(gamsel_lin_df, temp_df)

  ###Erstes Auftreten jeder Variable bestimmen (nicht-linear)

  beta = t(gamsel_fit$betas)
  min_lambda = apply(beta, 2,min_level)
  min_lambda[is.infinite(min_lambda)] = NA

  ###Dataframe erstellen (nicht-linear)

  temp_df = as.data.frame(t(min_lambda))
  colnames(temp_df) = beta_names
  temp_df = temp_df[, !duplicated(colnames(temp_df))]
  gamsel_non_lin_df = rbind.fill(gamsel_non_lin_df, temp_df)

}

###Dataframes abspeichern

save(gamsel_lin_df ,file="data/Processed Data/gamsel_lin.Rda")
save(gamsel_non_lin_df ,file="data/Processed Data/gamsel_non_lin.Rda")
```

**Code B.5:** Simulationstudie

```
library(dplyr)
library(dummies)
library(plyr)
library(data.table)
```

```
library(glinternet)
library(gglasso)
library(pliable)
library(gamsel)


###Dataframes vorbereiten

set.seed(123)

main_glinternet = data.frame()
int_glinternet = data.frame()

main_gglasso = data.frame()
int_gglasso = data.frame()

main_pliable = data.frame()
int_pliable = data.frame()

main_gamsel = data.frame()
int_gamsel = data.frame()
nonlin_gamsel = data.frame()


###Funktion um Daten zu erstellen

sim_data = function(n,beta0,beta1,beta2,beta3,theta1,theta2, noise){
  X1 = rnorm(n)
  X2 = rnorm(n)
  X3 = sample(seq(5), n, replace=TRUE, prob=rep(0.2,5))
  beta3 = c(beta3, beta3, -beta3, -beta3)
  theta2 = X1%*%t(c(theta2, theta2, -theta2, -theta2))

  ###Linearen Prädiktor generieren

  lin_pred = beta0 + beta1*X1 + beta2*X2 +
    dummy(X3)[, -1]%*%beta3 +
    theta1*X1*X2 +
    rowSums(dummy(X3)[, -1]*theta2) +
    rnorm(n)

  ###Umskalieren in Wahrscheinlichkeiten

  prob = 1/(1+exp(-lin_pred))

  ###Zielgröße mit Bernoullli-Verteilung simulieren

  Y = as.integer(rbinom(n,1,prob))

  ###Dataframe erstellen

  df = data.frame(Y,X1,X2,as.factor(X3))

  ###Störvariablen hinzufügen

  for (i in 1:noise){
    df = cbind(df, rnorm(n))
  }
```

```
  colnames(df) = (c("Y", "X1", "X2","X3", paste0("S", seq(noise))))
  return(df)
}

###Funktion um erstes Auftreten der Variable zu bestimmen

min_level = function(x){
  min(which(x!=0))
}

###Funktion um nur Einflussvariablen im finalen Dataframe zu speichern

set_dataframe = function(x, t, v){
  temp = stack(x[colnames(x) %like% "X"])
  temp$Typ = t
  temp$Verfahren = v
  return(temp)
}

###Iterationen

for (k in 1:100){
  print(paste0("Progress: ", round((k-1)*100/100), "%"))

  ###Effektstärke festlegen

  df = sim_data(1000,-1,.2,.2,.5,0,0,100)

  ###GLINTERNET

  X = df%>%select(-Y)
  X$X3 = as.numeric(X$X3)-1
  pairs = matrix(c(rep(1,dim(df)[2]-2), seq(2,dim(df)[2]-1)), ncol = 2)
  numLevels = c(1,1,5, rep(1,dim(df)[2]-4))
  glinternet_fit = glinternet(X=X, Y=df$Y, family = "binomial",
                              numLevels=numLevels,interactionPairs = pairs)

  coefs = coef(glinternet_fit)

  ###Koeffizienten erhalten

  maineffect_cat = c()
  maineffect_num = c()
  int_cat = c()
  int_num = c()
  for (i in 1:50){
    maineffect_cat = c(maineffect_cat, coefs[[i]]$mainEffects$cat)
    maineffect_num = c(maineffect_num, coefs[[i]]$mainEffects$cont)
    int_cat = c(int_cat, coefs[[i]]$interactions$catcont[,1])
    temp_num = c(coefs[[i]]$interactions$contcont[,1], coefs[[i]]$interactions$
↪contcont[,2])
    temp_num = temp_num[temp_num != 1]
    int_num = c(int_num, temp_num)
  }

  ####Namen erhalten

  num_names = names(df %>% select(-c(Y,X3)))
```

```
120
121   ####Erstes vorkommen von Variablen bestimmen
122
123   first_main_effect_cat = 50-table(maineffect_cat)
124   first_main_effect_num = 50-table(maineffect_num)
125   first_int_effect_cat = 50-table(int_cat)
126   first_int_effect_num = 50-table(int_num)
127
128   if (is.null(int_cat)){
129     first_int_effect_cat = NA
130   }
131   if (is.null(maineffect_cat)){
132     first_main_effect_cat = NA
133   }
134   names(first_main_effect_cat) = "X3"
135   names(first_int_effect_cat) = "X3"
136
137   ###Namen festlegen für die Variablen
138
139   main_effect_num_names = c()
140   int_effect_num_names = c()
141
142   for (i in names(first_main_effect_num)){
143     i = as.numeric(i)
144     main_effect_num_names = c(main_effect_num_names, num_names[i])
145   }
146   for (i in names(first_int_effect_num)){
147     i = as.numeric(i)
148     int_effect_num_names = c(int_effect_num_names, num_names[i])
149   }
150
151   names(first_main_effect_num) = main_effect_num_names
152   names(first_int_effect_num) =  int_effect_num_names
153
154   ###Dataframe für die Haupteffekte erstellen
155
156   first_main_effect_cat = data.frame(rbind(first_main_effect_cat))
157   first_main_effect_num = data.frame(rbind(first_main_effect_num))
158   main_df_temp = as.data.frame(rbind(t(first_main_effect_cat), t(first_main_effect_num
    ↪)))
159   main_df_temp = main_df_temp %>% dplyr::mutate(Rang = rank(first_main_effect_cat,
    ↪ties.method = "min"))
160   main_df_temp = as.data.frame(t(main_df_temp %>% select(-"first_main_effect_cat")))
161
162   ###Dataframe für die Interaktionen erstellen
163
164   first_int_effect_cat = data.frame(rbind(first_int_effect_cat))
165   first_int_effect_num = data.frame(rbind(first_int_effect_num))
166   int_df_temp = as.data.frame(rbind(t(first_int_effect_cat), t(first_int_effect_num)))
167   int_df_temp = int_df_temp %>% dplyr::mutate(Rang = rank(first_int_effect_cat, ties.
    ↪method = "min"))
168   int_df_temp = as.data.frame(t(int_df_temp %>% select(-"first_int_effect_cat")))
169
170   ###Dataframes zusammenführen
171
172   main_glinternet = rbind.fill(main_glinternet, main_df_temp)
173   int_glinternet = rbind.fill(int_glinternet, int_df_temp)
174
```

```
175  ###GGLasso trainieren
176
177  numLevels = c(1,1,4, rep(1,dim(df)[2]-4))
178  groups = c(1,2, rep(3,4), seq(4, dim(df)[2]), rep(dim(df)[2]+1, 4), seq(dim(df)[2]+2
     ↪, dim(df)[2]*2-3))
179  X = model.matrix(~ . + X1 * . , dplyr::select(df, -Y))[, -1]
180  Y = df$Y
181  Y[Y==0] = -1
182  group_lasso_fit = gglasso(x = X, y = Y, group = groups, nlambda = 50, loss = "logit"
     ↪)
183
184  ####Erstes vorkommen von Variablen bestimmen
185
186  beta = t(group_lasso_fit$beta)
187  min_lambda = apply(beta, 2,min_level)
188  min_lambda[is.infinite(min_lambda)] = NA
189
190  ###Dataframe für Haupteffekte erstellen
191
192  main_names = c()
193  for (n in colnames(df %>% select(-Y))){
194    if(n == "X3"){
195      main_names = c(main_names, rep(n, 4))
196    }
197    else
198      main_names = c(main_names, n)
199  }
200  main_df = min_lambda[-1:dim(df)[2]+2]
201  names(main_df) = main_names
202  main_df = main_df[!duplicated(names(main_df))]
203  main_df = as.data.frame(main_df)
204  main_df = main_df %>% dplyr::mutate(Rang = rank(main_df, ties.method = "min"))
205  main_df = as.data.frame(t(main_df %>% select(-"main_df")))
206
207  ###Dataframe für Interaktionseffekte erstellen
208
209  int_names = main_names[-1]
210  int_df = min_lambda[(dim(df)[2]+3):(2*dim(df)[2]+3)]
211  names(int_df) = int_names
212  int_df = int_df[!duplicated(names(int_df))]
213  int_df = as.data.frame(int_df)
214  int_df = int_df %>% dplyr::mutate(Rang = rank(int_df, ties.method = "min"))
215  int_df = as.data.frame(t(int_df %>% select(-"int_df")))
216
217  ###Dataframes zusammenführen
218
219  main_gglasso = rbind.fill(main_gglasso, main_df)
220  int_gglasso = rbind.fill(int_gglasso, int_df)
221
222  ###Pliable trainieren
223
224  X = model.matrix(~ . , dplyr::select(df, -c(Y,X1)))[, -1]
225  Y = df$Y
226  pliable_fit = pliable(X, z = df$X1, y = Y,
227                        family = "binomial")
228
229
230  ###Dataframe für Haupteffekte erstellen
```

```
231
232   beta = t(pliable_fit$beta)
233   min_lambda = apply(beta, 2,min_level)
234   min_lambda[is.infinite(min_lambda)] = NA
235   main_df = as.data.frame(t(min_lambda))
236   colnames(main_df) = colnames(X)
237   main_df = as.data.frame(t(cbind(main_df,data.frame(X1 = 1))))
238   main_df$names = row.names(main_df)
239   main_df$names[main_df$names %like% "X3"] = "X3"
240   main_df = main_df %>% group_by(names) %>% dplyr::summarise(V1 = min(V1))
241   main_df = main_df %>% dplyr::mutate(Rang = rank(V1, ties.method = "min"))
242   main_df = as.data.frame(t(main_df %>% select(-"V1")))
243   colnames(main_df) = main_df[1,]
244   main_df = main_df[2,]
245
246   ###Dataframe für Interaktionen erstellen
247
248   theta = t(pliable_fit$theta[,1,])
249   min_lambda = apply(theta, 2,min_level)
250   min_lambda[is.infinite(min_lambda)] = NA
251   int_df = as.data.frame(t(min_lambda))
252   colnames(int_df) = colnames(X)
253   int_df = as.data.frame(t(int_df))
254   int_df$names = row.names(int_df)
255   int_df$names[int_df$names %like% "X3"] = "X3"
256   int_df = int_df %>% group_by(names) %>% dplyr::summarise(V1 = min(V1))
257   int_df = int_df %>% dplyr::mutate(Rang = rank(V1, ties.method = "min"))
258   int_df = as.data.frame(t(int_df %>% select(-"V1")))
259   colnames(int_df) = int_df[1,]
260   int_df = int_df[2,]
261
262   ###Dataframes zusammenführen
263
264   main_pliable = rbind.fill(main_pliable, main_df)
265   int_pliable = rbind.fill(int_pliable, int_df)
266
267   ###GAMSEL trainieren
268
269   X = model.matrix(~ . + X1 * . , dplyr::select(df, -Y))[, -1]
270   Y = df$Y
271   degrees = c(10,10,rep(1,4),rep(10,(dim(df)[2]-4)),10,rep(1,4),rep(10,(dim(df)[2]-4))
    ↪)
272   gamsel_fit = gamsel(X,Y, family = "binomial", traceit = F, degrees = degrees)
273
274   ###Dataframe für Haupteffekte erstellen
275
276   alpha = t(gamsel_fit$alphas)
277   min_lambda = apply(alpha, 2,min_level)
278   min_lambda[is.infinite(min_lambda)] = NA
279   temp_df = as.data.frame(t(min_lambda))
280   colnames(temp_df) = colnames(X)
281   main_df = temp_df[-1:dim(df)[2]+2]
282   main_df = as.data.frame(t(main_df))
283   main_df$names = row.names(main_df)
284   main_df$names[main_df$names %like% "X3"] = "X3"
285   main_df = main_df %>% group_by(names) %>% dplyr::summarise(V1 = min(V1))
286   main_df = main_df %>% dplyr::mutate(Rang = rank(V1, ties.method = "min"))
287   main_df = as.data.frame(t(main_df %>% select(-"V1")))
```

```
  colnames(main_df) = main_df[1,]
  main_df = main_df[2,]

  ###Dataframe für Interaktionen erstellen

  int_df = temp_df[(dim(df)[2]+3):(dim(df)[2]*2+3)]
  int_df = as.data.frame(t(int_df))
  int_df$names = row.names(int_df)
  int_df$names[int_df$names %like% "X3"] = "X3"
  int_df = int_df %>% group_by(names) %>% dplyr::summarise(V1 = min(V1))
  int_df = int_df %>% dplyr::mutate(Rang = rank(V1, ties.method = "min"))
  int_df = as.data.frame(t(int_df %>% select(-"V1")))
  colnames(int_df) = int_df[1,]
  int_df = int_df[2,]
  colnames(int_df) = colnames(main_df %>% select(-X1))

  ###Dataframes zusammenführen

  main_gamsel = rbind.fill(main_gamsel, main_df)
  int_gamsel = rbind.fill(int_gamsel, int_df)
}


###Dataframes vorbereiten für die Evaluation

main_glinternet = set_dataframe(main_glinternet, "Haupteffekt", "glinternet")
main_gglasso = set_dataframe(main_gglasso, "Haupteffekt", "gglasso")
main_pliable = set_dataframe(main_pliable, "Haupteffekt", "pliable")
main_gamsel = set_dataframe(main_gamsel, "Haupteffekt", "gamsel")

int_glinternet = set_dataframe(int_glinternet, "Interaktionseffekt", "glinternet")
int_gglasso = set_dataframe(int_gglasso, "Interaktionseffekt", "gglasso")
int_pliable = set_dataframe(int_pliable, "Interaktionseffekt", "pliable")
int_gamsel = set_dataframe(int_gamsel, "Interaktionseffekt", "gamsel")

###Dataframes zusammenführen

result = rbind(main_glinternet, main_gglasso, main_pliable, main_gamsel,
               int_glinternet, int_gglasso, int_pliable, int_gamsel)

###Resultierende Tabelle abspeichern

save(result ,file="data/Processed Data/sim_only_weak_main.Rda")
```