Ludwig-Maximilians-Univeristät München
Institut für Statistik
Seminar für angewandte Stochastik
Prof. Dr. Gerhard Tutz

Masterarbeit

Regularisierungsmethoden zur Merkmalsauswahl in kategorialen Responsemodellen

Betreuung: Sebastian Petry (M.Sc.) und
Dipl.-Stat. Wolfgang Pößnecker
Verfasser: Lorenz Uhlmann

2. Mai 2011

Inhaltsverzeichnis

1 Einführung **3**

2 Kategoriale Responsemodelle **4**
2.1 Das binäre Logitmodell 4
2.1.1 Metrische Kovariablen 4
2.1.2 Kategoriale Kovariablen 5
2.1.3 Schätzung der Parameter 6
2.2 Das multinomiale Logitmodell 7
2.2.1 Globale Einflussgrößen - kategorienspezifische Koeffizienten 7
2.2.2 Globale Koeffizienten - kategorienspezifische Einflussgrößen 10
2.2.3 Nebenbedingungen 12

3 Regularisierung im Logitmodell **14**
3.1 Regularisierung von Parametern 14
3.2 Regularisierung von Kovariablen 15

4 Algorithmus zur Merkmalsauswahl **17**
4.1 Lasso 17
4.2 Group Lasso 17
4.3 Algorithmus für das binäre logistische Modell 18
4.4 Algorithmus für das multinomiale logistische Modell 19
4.4.1 Metrische Kovariablen 20
4.4.2 Kategoriale Kovariablen 23
4.4.3 Kategorienspezifische Kovariablen 24
4.5 Alternative Variablenselektion 28
4.6 Kombinierte Variablenselektion 30
4.7 Umsetzung des Algorithmus 33

5 Anwendung des Algorithmus **37**
5.1 Simulationsstudie 37
5.2 Reale Daten 54

6 Zusammenfassung und Ausblick **62**

Literatur **65**

Anhang A **66**

Anhang B **76**

1 Einführung

In dieser Arbeit soll ein Verfahren vorgestellt werden, welches Variablenselektion in kategorialen Responsemodellen ermöglicht. Ausgangspunkt ist also eine Datensituation, in welcher die Zielvariable in Kategorien und Kovariablen, welche auf Zusammenhangsstrukturen mit dem Response untersucht werden sollen, vorliegen. Aufgrund der (relativ) leichten Interpretierbarkeit der daraus resultierenden Koeffizientenwerte, finden hierbei oftmals multinomiale Logitmodelle Einsatz. Jedoch kann es hierbei passieren, dass die entsprechenden Schätzer nicht eindeutig definiert sind bzw. einige von ihnen gegen $\pm\infty$ gehen (vgl. Zahid und Tutz, 2009, S. 1). Bzgl. dieser Problematik wurden für die Schätzung der Koeffizienten bspw. bei Verwendung der linearen Regression bereits viele Verfahren vorgestellt und weiterentwickelt. Für das multinomiale Logitmodell sind solche „Penalisierungs"- bzw. „Regularisierungsverfahren" ebenfalls vorhanden, welche auch das einfache binäre oder multinomiale Logitmodell in Situationen ohne Schätzprobleme bzgl. des „mean squared error" (MSE) u.U. dominieren können (vgl. Zahid und Tutz, 2009, S. 10 ff.). Ein Nachteil dieser Methoden kann jedoch darin bestehen, dass Prädiktoren nur für einzelne Kategorien der Responsevariable regularisiert werden und somit i.A. keine direkte Variablenselektion durchgeführt wird. Um dieser Problematik entgegenzuwirken, soll in Kapitel 4.4 ein Algorithmus vorgestellt werden, anhand welchem Merkmalsauswahl möglich ist, Prädiktoren also für alle Responsekategorien oder gar nicht in das Modell aufgenommen und ihre Koeffizienten ggf. penalisiert werden. Hierbei findet prinzipiell ein multinomiales Logitmodell, welches in Kapitel 2.2 behandelt wird, Anwendung. Die entsprechende Log-Likelihood wird im Algorithmus über einen Strafterm aus dem Kontext des „Lasso" (Tibshirani, 1996) penalisiert. In der hier verwendeten Form ist es möglich, gesamte „nicht relevante" Kovariablen aus dem Modell zu nehmen, was ggf. zur Übersichtlichkeit beiträgt, und die verbleibenden gegen Null zu shrinken. Wie in Kapitel 5.1 gezeigt wird, wirkt sich dies in vielen Situation positiv bzgl. des MSE($\boldsymbol{\beta}$) und des MSE($\boldsymbol{\pi}$) (für entsprechende Definitionen vgl. Kapitel 5.1) auf die Schätzungen aus.

2 Kategoriale Responsemodelle

Im Folgenden soll die Grundstruktur der Modelle, auf welchen der später gezeigte Algorithmus aufbaut, erläutert werden. Hierbei wird zunächst vom Fall binärer Zielvariablen ausgegangen und die Methodik anschließend auf mehrkategorialen Response erweitert.

2.1 Das binäre Logitmodell

Bei Daten mit einer Zielvariable, welche lediglich zwei Kategorien besitzt, existieren mehrere Möglichkeiten zur Modellierung. Die Herangehensweise, die hier genauer behandelt wird, ist aus dem Kontext der generalisierten linearen Modelle unter Verwendung der Binomialverteilung mit der Logitfunktion als Linkfunktion ableitbar. Die Ausführungen des gesamten Kapitels 2.1 beziehen sich im Wesentlichen auf Fahrmeir et al. (2007, Kapitel 4).

2.1.1 Metrische Kovariablen

Wie bereits erwähnt, soll in diesem Abschnitt von binärem Response ausgegangen werden, wobei die Ausprägungen mit 0 und 1 kodiert werden. Sämtliche Kovariablenausprägungen seien hierbei zunächst ausschließlich von metrischem Skalenniveau und somit auch nicht weiter gruppierbar. Für die Daten ergibt sich damit

$$y_i \in \{0,1\} \quad \text{und} \quad \boldsymbol{x}_i \in \mathbb{R}^p, \quad i = 1,\dots,n,$$

mit n als Anzahl der Beobachtungen und p als Anzahl der Einflussgrößen, wobei im Folgenden stets davon ausgegangen wird, dass sowohl $n > 1$, als auch $p > 1$ gilt. Die Auftretenswahrscheinlichkeit von $Y_i^\star = 1$ gegeben x_i

$$\mathbb{P}(Y_i^\star = 1|\boldsymbol{x}_i) = \pi_i, \quad i = 1,\dots,n,$$

wobei $Y_i^\star$ die zu y_i $(i = 1,\dots,n)$ gehörende Zufallsvariable bezeichnet, soll nun anhand des linearen Prädiktors

$$\eta_i = \beta_0 + x_{i1}\beta_1 + \ldots + x_{ip}\beta_p, \quad i = 1,\dots,n$$

über die Beziehung

$$\pi_i = \frac{\exp(\eta_i)}{1+\exp(\eta_i)}, \quad i = 1,\dots,n \tag{1}$$

modelliert werden. β_j $(j = 1,\dots,p)$ stellt hierbei den Koeffizienten zur entsprechenden Kovariable dar, während mit β_0 der Intercept bezeichnet wird. Für die Gegenwahrscheinlichkeit ergibt sich daraus

$$1 - \pi_i = 1 - \frac{\exp(\eta_i)}{1+\exp(\eta_i)} = \frac{1}{1+\exp(\eta_i)}, \quad i = 1,\dots,n.$$

Durch Umstellung der Gleichung (1) kann auch die Darstellung

$$\log(\frac{\pi_i}{1-\pi_i}) = \eta_i, \quad i = 1, \ldots, n \tag{2}$$

bzw.

$$\frac{\pi_i}{1-\pi_i} = \exp(\eta_i), \quad i = 1, \ldots, n,$$

welche den Zusammenhang des Chancenverhältnisses zwischen den beiden Kategorien zum linearen Prädiktor verdeutlicht, gewonnen werden.

2.1.2 Kategoriale Kovariablen

Für eine Erweiterung auf nominalskalierte Kovariablen ist die Datenmatrix entsprechend anzupassen. Die folgenden Ausführungen zu den Kodierungsmöglichkeiten beziehen sich auf Fahrmeir et al. (2007, S. 80 ff.). Geht man von G Kovariablen aus, welche entweder von metrischem oder nominalem Skalenniveau sind, so kann jede Variable g, falls sie kategorial ist, bspw. durch df_g (Anzahl ihrer Kategorien - 1) ($g = 1, \ldots, G$) Dummy-Variablen ersetzt werden. Dabei muss eine Referenzkategorie gewählt werden, auf welche sich die jeweils dazugehörigen 0/1-kodierten Kovariablen beziehen. Hierfür wird hier die jeweils erste Kategorie verwendet. Somit ergibt sich bspw. für eine Einflussgröße g mit insgesamt fünf möglichen Kategorien $(1, 2, 3, 4, 5)$ und der Auprägung $\boldsymbol{x}_g = (1, 1, 2, 2, 3, 4, 5, \ldots)$

$$\boldsymbol{X}_g = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}.$$

Sind alle Dummy-Variablen gleich Null, so stammt die Beobachtung aus der ersten Kategorie (der Referenzkategorie) der Kovariable. Steht in der ersten Spalte eine Eins, steht im Originaldatensatz eine Zwei usw.. Pro Zeile darf damit natürlich nur eine Eins zu finden sein. Die Nummerierung der Kategorien hat also keinerlei inhaltliche Bedeutung, sondern dient lediglich der Kodierung. In Fahrmeir et al. (2007, S. 80 ff.) ist eine ausführlichere Darstellung der Dummy-Kodierung zu finden.

Eine alternative Herangehensweise stellt die Effekt-Kodierung dar. Geht man von einer einzelnen kategorialen Kovariable g mit df_g Kategorien aus, so wird aus dem Kovariablenvektor die entsprechende Matrix $\boldsymbol{X}_g = (\boldsymbol{x}_{g1}, \ldots, \boldsymbol{x}_{g,\text{df}-1})$. Die

einzelnen Vektoren werden dabei über $\boldsymbol{x}_{ig} = (x_{ig1}, \ldots, x_{ig,\mathrm{df}_g-1})$ mit

$$x_{ig1} = \begin{cases} 1 & \text{falls } x_{ig} = 2 \\ -1 & \text{falls } x_{ig} = 1 \\ 0 & \text{sonst} \end{cases}, \quad \ldots, \quad x_{ig,\mathrm{df}_g-1} = \begin{cases} 1 & \text{falls } x_{ig} = \mathrm{df}_g \\ -1 & \text{falls } x_{ig} = 1 \\ 0 & \text{sonst} \end{cases}$$

konstruiert (vgl. Fahrmeir et al., 2007, S. 82 f.). Das Komma in $\boldsymbol{x}_{g,\mathrm{df}-1}$ dient lediglich der besseren Lesbarkeit.

Während in Fahrmeir et al. (2007, S. 82 f.) die df_g-te Kategorie mit -1 kodiert wird, wird hier die jeweils erste verwendet. Diese Kodierung findet auch in dem später gezeigten Algorithmus und in der Funktion „`plregr`“ (vgl. Anhang A bzw. das Dokument „Algorithmus.r“ auf der beigefügten CD), in welcher das in Kapitel 4.6 vorgestellte Verfahren umgesetzt wurde, Verwendung. Für den oben gezeigten Beispielvektor $\boldsymbol{x}_g = (1, 1, 2, 2, 3, 4, 5, \ldots)$ gilt in dieser Kodierung

$$\boldsymbol{X}_g = \begin{pmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}.$$

Es sei noch erwähnt, dass im Folgenden stets auch $G > 1$ angenommen wird.

Ist ein Prädiktor von metrischem Skalenniveau, so wird nichts verändert und es gilt $\mathrm{df}_g = 1$ $(g = 1, \ldots, G)$. Definiert man nun $\sum_{g=1}^{G} \mathrm{df}_g = p$, so ist die oben gezeigte Modellierung mit der umformulierten Datenmatrix direkt anwendbar. Hierbei ist auch sofort ersichtlich, dass in einem Datensatz mit ausschließlich metrischen Einflussgrößen $G = p$ gilt. In solchen Situationen wird im Folgenden die Anzahl der Einflussgrößen stets mit p bezeichnet, während die einzelnen Prädiktoren mit j $(j = 1, \ldots, p)$ indiziert werden.

2.1.3 Schätzung der Parameter

Zur Schätzung der Parameter kann das Maximum-Likelihood (ML)-Verfahren Anwendung finden. Hierbei wird in x_i eine Eins für den Intercept aufgenommen, sodass von nun an

$$x_i = (1, x_{i1}, \ldots, x_{x_ip})$$

gilt. Außerdem sei

$$\boldsymbol{\beta} = (\beta_0, \ldots, \beta_p).$$

Die Likelihood hierfür ergibt sich zu

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} L_i(\boldsymbol{\beta}) = \prod_{i=1}^{n} f(y_i|\pi_i), \tag{3}$$

da die y_i $(i = 1, \ldots, n)$ als bedingt unabhängig angenommen werden. Über den oben gezeigten Zusammenhang hängt π_i wiederum von den zu schätzenden Parametern in $\boldsymbol{\beta}$ ab. Nach Einsetzen der Dichte der entsprechenden Binomialverteilung $(y_i \sim B(1, \pi_i))$

$$f(y_i|\boldsymbol{\beta}; \boldsymbol{x}_i) = \pi_i^{y_i}(1 - \pi_i)^{1-y_i}$$

erhält man für die Likelihood

$$L(\beta) = \prod_{i=1}^{n} \pi_i^{y_i}(1 - \pi_i)^{1-y_i}.$$

Durch Logarithmieren ergibt sich daraus die Log-Likelihood

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} l_i(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left\{ y_i \log\left(\frac{\pi_i}{1 - \pi_i}\right) + \log(1 - \pi_i) \right\} \tag{4}$$

bzw. nach Einsetzen (und Umstellen) von Gleichung (2)

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} y_i(\boldsymbol{x}_i^T \boldsymbol{\beta}) - \log(1 + \exp(\boldsymbol{x}_i^T \boldsymbol{\beta})),$$

welche es zu maximieren gilt, was bspw. über das Fisher-Scoring-Verfahren oder das Newton-Raphson-Verfahren möglich ist (vgl. Fahrmeir et al., 2007, S. 473 ff.).

2.2 Das multinomiale Logitmodell

Eine Verallgemeinerung des für binären Response behandelten Modells ist der Übergang zu mehrkategorialen Zielvariablen. Auch in den Kapiteln 2.2.1 und 2.2.2 beziehen sich die Ausführungen im Wesentlichen auf Fahrmeir et al. (2007, Kapitel 5).

2.2.1 Globale Einflussgrößen - kategorienspezifische Koeffizienten

In Kapitel 2.1.1 wurde π_i als Wahrscheinlichkeit $\mathbb{P}(Y_i^\star = 1|\boldsymbol{x}_i)$ $(i = 1, \ldots, n)$ eingeführt, welche über Prädiktoren modelliert wird. Wenn man nun zu einer Responsevariable mit $K > 2$ Kategorien übergeht, so kann das im binären Fall skalare π_i zu einem Vektor der Länge $K - 1 = q$ umgeformt werden, worin nun die einzelnen Wahrscheinlichkeiten, dass die Beobachtung i aus einer bestimmten Responsekategorie bei gegebenem $\boldsymbol{x}_i$ stammt, enthalten sind. Die „letzte“ Wahrscheinlichkeit (π_{iK}) ergibt sich dabei aus den q Einträgen von $\boldsymbol{\pi}_i$. Diese Wahrscheinlichkeiten können nun modelliert werden, indem eine Referenzkategorie gewählt und diese als „Gegenereignis“ verwendet wird, für welche im Folgenden die K-te Kategorie verwendet wird. Entsprechend wird nun für jede Kategorie, mit Ausnahme der Referenzkategorie, ein eigener Koeffizientenvektor

$\boldsymbol{\beta}_{r.} = (\beta_{r0}, \ldots, \beta_{rp})$ $(r = 1, \ldots, q)$ benötigt, welcher in einer zum binären Fall analogen Weise mit der jeweiligen Responsekategorie verknüpft wird. Im Folgenden wird ausschließlich von metrischen Kovariablen ausgegangen. Für die Wahrscheinlichkeiten ergibt sich also

$$\pi_{ir} = \frac{\exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.})}{1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})}, \quad r = 1, \ldots, q, \quad i = 1, \ldots, n \tag{5}$$

bzw.

$$\pi_{iK} = 1 - \pi_{i1} - \ldots - \pi_{iq} = \frac{1}{1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})}, \quad i = 1, \ldots, n$$

mit

$$\pi_{ir} = \mathbb{P}(Y_i^\star = r | \boldsymbol{x}_i) \quad i = 1, \ldots, n,$$

wobei $Y_i^\star$ die Zufallsvariable zum Response bezeichnet. Durch Umformulierung erhält man die Beziehung zwischen dem logarithmierten Chancenverhältnis und dem jeweiligen linearen Prädiktor als

$$\log \frac{\pi_{ir}}{\pi_{iK}} = \boldsymbol{x}_i^T \boldsymbol{\beta}_{r.}, \quad r = 1, \ldots, q, \quad i = 1, \ldots, n.$$

Analog zum binären Fall können die Parameter auch bei multinomialen Logitmodellen über das ML-Verfahren geschätzt werden. Hierbei gilt für die Zielvariable

$$\boldsymbol{y}_i | \boldsymbol{x}_i \sim M(1, \boldsymbol{\pi}_i),$$

wobei $M(\cdot, \cdot)$ die Multinomialverteilung beschreibt. Diese kann als direkte Verallgemeinerung der Binomialverteilung angesehen werden. Man nehme für eine Variable $\boldsymbol{y}^\diamond \sim M(m, \boldsymbol{\pi})$, mit m unabhängigen Wiederholungen, wobei

$$\boldsymbol{y}^\diamond = \begin{pmatrix} y_1^\diamond \\ \vdots \\ y_q^\diamond \end{pmatrix}$$

gilt, an. Die einzelnen Einträge von $\boldsymbol{y}^\diamond$ beschreiben die Häufigkeiten, mit welchen die jeweiligen Kategorien gezogen wurden. Hierbei ist die entsprechende Dichte definiert als

$$\begin{aligned} f(\boldsymbol{y}^\diamond | \boldsymbol{\pi}) = \frac{m!}{y_1^\diamond! \cdot \ldots \cdot y_q^\diamond! (m - y_1^\diamond - \ldots - y_q^\diamond)!} \pi_1^{y_1^\diamond} \cdot \ldots \\ \cdot \pi_q^{y_q^\diamond} (1 - \pi_1 - \ldots - \pi_q)^{1 - y_1^\diamond - \ldots - y_q^\diamond}. \end{aligned}$$

Für die Schätzung der Parameter in einem multinomialen Logitmodell wird angenommen, dass die Zielvariablen $\boldsymbol{y}_i$ $(i = 1, \ldots, n)$ (bedingt) unabhängig und nach $\boldsymbol{y}_i | \boldsymbol{x}_i \sim M(1, \boldsymbol{\pi})$ (bzw. allgemein bei gruppierten Daten nach $M(n_i, \boldsymbol{\pi}_i)$, mit n_i

als die Anzahl der Beobachtungen mit einer bestimmten Kombination an Kovariablenausprägungen) verteilt sind. Ein $\boldsymbol{y}_i$ beschreibt nun einen Vektor der Länge q entsprechend den Responsekategorien, ohne Berücksichtigung der Referenzkategorie. Stammt die Beobachtung i aus der r-ten Kategorie, so enthält $\boldsymbol{y}_i$ an der r-ten Stelle eine Eins und sonst Nullen. Stammt die Beobachtung aus der K-ten Kategorie, so enthält $\boldsymbol{y}_i$ ausschließlich Nullen. Damit lässt sich die Likelihood schreiben als

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} f(\boldsymbol{y}_i|\boldsymbol{\pi}_i)$$

Durch Logarithmieren der Likelihood und Vernachlässigung des von $\boldsymbol{\beta}$ unabhängigen Terms erhält man

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n}(y_{i1}\log\pi_{i1} + \ldots + y_{iq}\log\pi_{iq} + y_{iK}\log\pi_{iK}), \tag{6}$$

mit $y_{iK} = 1 - y_{i1} - \ldots - y_{iq}$ und $\pi_{iK} = 1 - \pi_{i1} - \ldots - \pi_{iq}$.

Für eine kompaktere Darstellung können nun sämtliche $\boldsymbol{\beta}_{r.}$ $(r = 1, \ldots, q)$ zu einem Vektor zusammengefasst werden, indem sie zu

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\beta}_{1.} \\ \vdots \\ \boldsymbol{\beta}_{q.} \end{pmatrix} \tag{7}$$

direkt untereinandergeschrieben werden. Zur Verwendung dieses Vektors muss die Datenmatrix

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_n \end{pmatrix} \tag{8}$$

der Dimension $n \cdot q \times (p+1) \cdot q$ mit

$$\boldsymbol{X}_i = \begin{pmatrix} \boldsymbol{x}_i^T & & & \\ & \boldsymbol{x}_i^T & & \\ & & \ddots & \\ & & & \boldsymbol{x}_i^T \end{pmatrix}, \quad i = 1, \ldots, n$$

zusammengefasst werden. Damit lässt sich der lineare Prädiktor schreiben als

$$\boldsymbol{\eta}_i = \boldsymbol{X}_i\boldsymbol{\beta}, \quad i = 1, \ldots, n, \tag{9}$$

bzw.

$$\boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}.$$

Außerdem kann noch der Vektor

$$\boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_n \end{pmatrix}$$

der Länge $n \cdot q$ gebildet werden.

Durch Ableiten der Log-Likelihood kann die Scorefunktion

$$l'(\boldsymbol{\beta}) = \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{\pi})$$

gewonnen werden. Mit

$$\hat{\boldsymbol{A}} = F^{-1}(\hat{\boldsymbol{\beta}}) = (E(l'(\hat{\boldsymbol{\beta}}) l'^T(\hat{\boldsymbol{\beta}})))^{-1}$$

gilt unter bestimmten Regularitätsannahmen

$$\hat{\boldsymbol{\beta}} \overset{a}{\sim} N(\boldsymbol{\beta}, \hat{\boldsymbol{A}}).$$

Um auf die strukturellen Unterschiede zum folgenden Kapitel 2.2.2 hinzuweisen, sei die Art des Einflusses der Prädiktoren auf den Response nochmals kurz dargestellt. Die hier vorgestellten Kovariablenausprägungen sind pro Individuum und Prädiktor eindeutig festgelegt und haben für alle Responsekategorien dieselben Werte. Es handelt sich also um globale Einflussgrößen. Die Koeffizienten besitzten dagegen i.A. für jede Responsekategorie verschiedene Werte und sind somit kategorienspezifisch.

Für die Anwendung von kategorialen Variablen mit nominalem Skalenniveau kann völlig analog zum oben aufgeführten binären Fall vorgegangen werden, indem die entsprechenden Prädiktoren in Effekt-Kodierung eingebracht werden.

2.2.2 Globale Koeffizienten - kategorienspezifische Einflussgrößen

Eine etwas spezielle Erweiterung der Datenmatrix kann durch die Einführung von kategorienspezifischen Prädiktoren geschehen, welche mit w_{irl} ($i = 1, \ldots, n$, $r = 1, \ldots, K$ und $l = 1, \ldots, L$) bezeichnet und über die globalen Koeffizienten γ_l ($l = 1, \ldots, L$) in das Modell eingebracht werden. Jede kategorienspezifische Einflussgröße l ($l = 1, \ldots, L$) besitzt dabei pro Beobachtung und Responsekategorie i.A. unterschiedliche Ausprägungen. Pro Kovariable wird im Modell dagegen nur ein globaler Koeffizient angepasst. Die Datenmatrix pro Beobachtung ist somit zu

$$\boldsymbol{X}_i = \begin{pmatrix} \boldsymbol{x}_i^T & & & \boldsymbol{w}_{i1}^T - \boldsymbol{w}_{iK}^T \\ & \boldsymbol{x}_i^T & & \boldsymbol{w}_{i2}^T - \boldsymbol{w}_{iK}^T \\ & & \ddots & \vdots \\ & & \boldsymbol{x}_i^T & \boldsymbol{w}_{iq}^T - \boldsymbol{w}_{iK}^T \end{pmatrix}, \quad i = 1, \ldots, n,$$

und der lineare Prädiktor pro Beobachtung und Responsekategorie zu

$$\eta_{ir} = \boldsymbol{x}_i^T \boldsymbol{\beta}_{r.} + (\boldsymbol{w}_{ir} - \boldsymbol{w}_{iK})^T \boldsymbol{\gamma} \quad i = 1, \ldots, n, \quad r = 1, \ldots, q, \tag{10}$$

mit $\gamma = (\gamma_1, \ldots, \gamma_L)^T$, zu erweitern. $\boldsymbol{w}_{ir}$ setzt sich dabei aus den einzelnen Ausprägungen der kategorienspezifischen Kovariable für die Beobachtung i ($i = 1, \ldots, n$) und die Kategorie r ($r = 1, \ldots, q$) zusammen. Da in dem hier vorgestellten Modell eine Referenzkategorie verwendet wird, ist jeder kategorienspezifische Vektor der Ausprägungen ($\boldsymbol{w}_{ir}$, $r = 1, \ldots, q$, $i = 1, \ldots, n$) von dem der Referenzkategorie ($\boldsymbol{w}_{iK}$, $i = 1, \ldots, n$) zu subtrahieren. Für diese Variablen wird angenommen, dass diese von metrischem Skalenniveau sind. Fügt man nun $\boldsymbol{\beta}$ und $\boldsymbol{\gamma}$ in einem gemeinsamen Koeffizientenvektor $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1.}, \ldots, \boldsymbol{\beta}_{q.}, \boldsymbol{\gamma})$ zusammen, so lässt sich der lineare Prädiktor pro Beobachtung (η_i), unter Verwendung der erweiterten Datenmatrix $\boldsymbol{X}$, in der bereits eingeführten Schreibweise (Gleichung (9)) darstellen.

Die Schätzung der Parameter kann über die Verwendung der hier eingeführten Datenmatrix $\boldsymbol{X}$ und dem ergänzten Koeffizientenvektor $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1.}^T, \ldots, \boldsymbol{\beta}_{q.}^T, \boldsymbol{\gamma}^T)$ ebenfalls über das ML-Verfahren stattfinden. Als Log-Likelihood wird völlig analog zu Gleichung (6)

$$l(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} (y_{i1} \log \pi_{i1} + \ldots + y_{iq} \log \pi_{iq} + y_{iK} \log \pi_{iK}), \tag{11}$$

mit

$$\begin{aligned} \pi_{ir} &= \frac{\exp(\eta_{ir})}{1 + \sum_{s=1}^{q} \exp(\eta_{is})} \\ &= \frac{\exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.} + (\boldsymbol{w}_{ir} - \boldsymbol{w}_{iK})^T \boldsymbol{\gamma})}{1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.} + (\boldsymbol{w}_{is} - \boldsymbol{w}_{iK})^T \boldsymbol{\gamma})}, \quad i = 1, \ldots, n, \quad r = 1, \ldots, q, \end{aligned}$$

und

$$\pi_{iK} = 1 - \pi_{i1} - \ldots - \pi_{iq}, \quad i = 1, \ldots, n,$$

verwendet. Stammt eine Beobachtung i aus der Kategorie r so kann der entsprechende Summand geschrieben werden als

$$\begin{aligned} l_i(\boldsymbol{\beta}, \boldsymbol{\gamma}) &= \eta_{ir} - \log(1 + \sum_{s=1}^{q} \exp(\eta_{is})) \\ &= \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.} + (\boldsymbol{w}_{ir} - \boldsymbol{w}_{iK})^T \boldsymbol{\gamma}) \\ &\quad - \log(1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.} + (\boldsymbol{w}_{is} - \boldsymbol{w}_{iK})^T \boldsymbol{\gamma}) \end{aligned}$$

Zu beachten ist, dass in den weiteren Ausführungen in $\boldsymbol{\beta}$ der γ-Vektor ausgeschlossen wird und separat behandelt wird. Die hier eingeführte Schreibweise sollte lediglich der Vereinfachung und Veranschaulichung dienen. Es gilt also von nun an wieder $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1.}^T, \ldots, \boldsymbol{\beta}_{q.}^T)$.

2.2.3 Nebenbedingungen

Ein Thema, auf welches in Zahid und Tutz (2009, S. 2 ff.), worauf sich auch die folgenden Ausführungen beziehen, etwas näher eingegangen wird, ist das der Wahl einer akkuraten Nebenbedingung im multinomialen logistischen Modell. Die allgemeine Modellformulierung lautet

$$\pi_{ir} = \frac{\exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.})}{\sum_{s=1}^{K} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})}, \quad i = 1, \ldots, n \quad r = 1, \ldots, K. \tag{12}$$

Im Nenner wird also über alle Kategorien des Response summiert. In dieser Form treten allerdings Identifizierbarkeitsprobleme auf, was bedeutet, dass sich durch eine Addition der Koeffizienten mit einer Konstante dieselben geschätzten Wahrscheinlichkeiten ergeben würden. Um das Modell also überhaupt schätzen zu können muss eine Nebenbedingung eingeführt werden. Ein Beispiel hierfür ist das Festsetzen von

$$\boldsymbol{\beta}_{K.} = \mathbf{0},$$

was der Wahl von Kategorie K als Referenzkategorie enspricht. Durch einfaches Einsetzen dieser Nebenbedingung in die Gleichung (12) führt, da somit $\exp(\boldsymbol{x}_i \boldsymbol{\beta}_{K.}) = 1$ gilt, zu

$$\pi_{ir} = \frac{\exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.})}{1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})}, \quad r = 1, \ldots, q \quad i = 1, \ldots, n, \tag{13}$$

der oben eingeführten Modellformulierung mit Referenzkategorie, für welche natürlich nicht nur die K-te, sondern auch jede andere Responsekategorie gewählt werden kann.

Jedoch ist dies nicht die einzige Möglichkeit, Eindeutigkeit der Schätzer sicherzustellen. Alternativ kann auch

$$\sum_{s=1}^{K} \boldsymbol{\beta}_{s.} = \mathbf{0}$$

festgelegt und damit eine symmetrische Nebenbedingung verwendet werden.

Unterschiede ergeben sich nun bei der Interpretation der Parameter. Bei der Wahl ersterer Bedingung beschreiben die Parameter den Einfluss auf die logarithmierten Chancenverhältnisse gegenüber der Referenzkategorie. Bei der Wahl einer symmetrischen Nebenbedingung wird durch die Umstellung von Gleichung (12) zu

$$\frac{\pi_{ir}}{GM(\boldsymbol{x}_i)} = \frac{\exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.})}{\sqrt[K]{\prod_{s=1}^{K} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})}}, \quad r = 1, \ldots, K, \quad i = 1, \ldots, n \tag{14}$$

bzw.

$$\log\left(\frac{\pi_{ir}}{GM(\boldsymbol{x}_i)}\right) = \boldsymbol{x}_i^T \boldsymbol{\beta}_{r.}, \quad r = 1, \ldots, K, \quad i = 1, \ldots, n,$$

mit $GM(\boldsymbol{x}_i)$ als das geometrische Mittel über die Responsekategorien für eine Beobachtung x_i, deutlich, dass die Interpretation mit einer „mittleren“ Responsekategorie als Referenzkategorie völlig analog zu oben erfolgen kann. Es gilt außerdem für beide Formulierungen

$$\log\left(\frac{\pi_{ir}}{\pi_{is}}\right) = \boldsymbol{x}_i^T(\boldsymbol{\beta}_{r.} - \boldsymbol{\beta}_{s.}).$$

Klarerweise ergeben sich je nach Wahl der Nebenbedingung unterschiedliche Koeffizientenwerte. Seien im Folgenden $\boldsymbol{\beta}_{.j}$ $(j = 1, \ldots, p)$ die Koeffizienten aus einem Modell mit Referenzkategorie und $\boldsymbol{\beta}^{\star}_{.j}$ $(j = 1, \ldots, p)$ diejenigen aus einem Modell mit symmetrischer Nebenbedingung. Der Punkt wird hierbei lediglich zur Verdeutlichung der Unterscheidung zwischen der Aufteilung nach Referenzkategorie und Parameter bzgl. der Einflussgrößen zusätzlich eingefügt. Analog wird in späteren Abschnitten, in welchen ein multinomiales Modell behandelt wird, die Schreibweise $\boldsymbol{\beta}_{.g}$ verwendet. Beide Darstellungen lassen sich damit über die Gleichung

$$\begin{aligned}\boldsymbol{\beta}^{\star}_{.j} &= \boldsymbol{T}\boldsymbol{\beta}_{.j} \\ &= \begin{pmatrix} \frac{K-1}{K} & -\frac{1}{K} & \cdots & -\frac{1}{K} & -\frac{1}{K} \\ -\frac{1}{K} & \frac{K-1}{K} & \cdots & -\frac{1}{K} & -\frac{1}{K} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\frac{1}{K} & -\frac{1}{K} & \cdots & \frac{K-1}{K} & -\frac{1}{K} \\ -\frac{1}{K} & -\frac{1}{K} & \cdots & -\frac{1}{K} & \frac{K-1}{K} \end{pmatrix} \boldsymbol{\beta}_{.j}, \quad j = 1, \ldots, p,\end{aligned}$$

bzw.

$$\begin{aligned}\boldsymbol{\beta}_{.j} &= \boldsymbol{T}^{-1}\boldsymbol{\beta}^{\star}_{.j} \\ &= \begin{pmatrix} 2 & 1 & \ldots & 1 & 1 \\ 1 & 2 & \ldots & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \ldots & 2 & 1 \\ 1 & 1 & \ldots & 1 & 2 \end{pmatrix} \boldsymbol{\beta}^{\star}_{.j}, \quad j = 1, \ldots, p,\end{aligned}$$

ineinander überführen. Für Details sei hier auf Zahid und Tutz (2009, S. 2 ff.) verwiesen.

3 Regularisierung im Logitmodell

Das im vorherigen Abschnitt vorgestellte multinomiale Logitmodell ist laut Zahid und Tutz (2009, S. 1) bei mehrkategorialem Response das am häufigsten verwendete Modell. Bei hochdimensionalen Datensituationen ist jedoch nicht garantiert, dass die ML-Schätzer existieren. Abhilfe schaffen hier Regularisierungsmethoden, bei welchen auch bei eben dieser Hochdimensionalität Schätzungen für die Koeffizienten existieren (vgl. Zahid und Tutz, 2009, S. 7). Solche Regularisierungsverfahren verwenden bei der Maximierung der Likelihood bzw. Minimierung der negativen Log-Likelihood zusätzlich einen Penalisierungsterm. Die allgemeine Form lautet

$$l_p(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - \lambda J(\boldsymbol{\beta}) \tag{15}$$

(vgl. Zahid und Tutz, 2009, S. 7). Dabei ist $J(\boldsymbol{\beta})$ ein Penalisierungsterm, welcher je nach Verfahren entsprechend gewählt wird.

3.1 Regularisierung von Parametern

Für verschiedene Definitionen von $J(\boldsymbol{\beta})$ wurden bereits Methoden entwickelt. In Zahid und Tutz (2009), worauf sich die folgenden Darstellungen beziehen, wird die Ridge-Penalisierung verwendet, welche für lineare Modelle von Hoerl und Kennard (1970) eingeführt wurde (vgl. Zahid und Tutz, 2009, S. 7 f.). Bei binärem Response ist $J(\boldsymbol{\beta}) = \sum_{j=1}^{p} \beta_j^2$ und bei multikategorialem $J(\boldsymbol{\beta}) = \sum_{r=1}^{q} \sum_{j=1}^{p} \beta_{rj}^2$. Insgesamt ergibt sich für die penalisierte Log-Likelihood im binären Fall

$$l_p(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - \lambda \sum_{j=1}^{p} \beta_j^2 \tag{16}$$

und mit $K > 2$ Kategorien im Response bei der Modelldefinition mit symmetrischer Nebenbedingung

$$l_p(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - \lambda \sum_{r=1}^{K} \sum_{j=1}^{p} \beta_{rj}^2 \tag{17}$$

(vgl. Zahid und Tutz, 2009, S. 7 f.). In Zahid und Tutz (2009) wird anstatt λ jeweils $\frac{\lambda}{2}$ verwendet (S. 7 ff.), woraus sich jedoch verständlicherweise äquivalente Schätzungen der Koeffizienten ergeben. Aus diesen Formulierungen ist zu erkennen, dass bei der Maximierung der jeweiligen Log-Likelihood die Größe der Koeffizienten durch die Summe über die quadrierten Werte bestraft wird und somit ein Trade-off zwischen Modellanpassung und -komplexität entsteht.

In einer Simulationsstudie wurde in Zahid und Tutz (2009, S. 10 ff.) gezeigt, dass dieses Verfahren in vielen Datensituationen den ML-Schätzer bzgl. des $\text{MSE}(\hat{\boldsymbol{\beta}})$ und $\text{MSE}(\hat{\boldsymbol{\pi}})$ (vgl. Zahid und Tutz, 2009, S. 12 ff., bzw. Kapitel 5.1 für entsprechende Definitionen) dominiert. Zudem beachte man, dass, falls $(p+1)\cdot K$,

also die Anzahl der gesamten zu schätzenden Koeffizienten, im Verhältnis zu n zu groß wird, beim ML-Schätzer Konvergenzprobleme auftreten können (vgl. Zahid und Tutz, 2009, S. 10 ff.).

Eine weitere Möglichkeit der Regularisierung wurde im Paper von Friedman et al. (2010), auf welches sich die folgenden Ausführungen beziehen, vorgestellt. Hierbei findet für eine Katorie r der Strafterm

$$\begin{aligned} J(\boldsymbol{\beta}_{r.}) &= \sum_{j=1}^{p} [\frac{1}{2}(1-\alpha)\beta_{rj}^2 + \alpha|\beta_{rj}|] \\ &= (1-\alpha)\frac{1}{2}\|\boldsymbol{\beta}_{r.}\|_2^2 + \alpha\|\boldsymbol{\beta}_{r.}\|_1, \quad r = 1, \ldots, K, \end{aligned} \tag{18}$$

mit $\|\boldsymbol{\beta}_{r.}\|_2 = (\sum_{j=1}^{p} \beta_{rj}^2)^{1/2}$ und $\|\boldsymbol{\beta}_{r.}\|_1 = \sum_{j=1}^{p} |\beta_{rj}|$, aus dem Kontext des „Elastic Net“ (vgl. Zou und Hastie, 2005) Verwendung. Die Koeffizienten werden also kategorienweise angepasst und es wird auch über alle Kategorien optimiert, da hierbei eine symmetrische Nebenbedingung Anwendung findet. Es wird also die allgemeine Modellformulierung aus Gleichung (12) für die Berechnungen herangezogen. Dabei ist dann das Maximierungsproblem

$$\max_{\boldsymbol{\beta}} \left[\frac{1}{n} l(\boldsymbol{\beta}) - \lambda \sum_{r=1}^{K} J_\alpha(\boldsymbol{\beta}_{r.}) \right], \tag{19}$$

wobei durch $J_\alpha(\cdot)$ die Abhängigkeit von α in Gleichung (19) verdeutlicht wird. Der erste Term von Gleichung (19) kann dann noch in der Form der Log-Likelihood als

$$l(\boldsymbol{\beta}) = \frac{1}{n} \sum_{i=1}^{n} \left[\sum_{r=1}^{K} y_{ir}(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.}) - \log \left(\sum_{r=1}^{K} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{r.}) \right) \right] \tag{20}$$

geschrieben werden.

Der hierbei verwendete Strafterm (18) ist eine Kombination aus Lasso- und Ridge-Penalisierung (für eine Darstellung des Lasso-Verfahrens s. Kapitel 4.1). Somit ist es im Gegensatz zur Anwendung der Ridge-Regression möglich, dass Parameter auf den Wert Null geschätzt werden und somit aus dem Modell genommen werden.

Als Ergänzung sei hier noch erwähnt, dass die oben eingeführte Nebenbedingung $\sum_{s=1}^{K} \boldsymbol{\beta}_{s.} = \mathbf{0}$ nicht exakt in dieser Form verwendet wird. Vielmehr wird die Uneindeutigkeit der Schätzer genutzt, um die penalisierte Log-Likelihood nochmals zu maximieren. Für Details sei auf Friedman et al. (2010, S. 11) verwießen.

3.2 Regularisierung von Kovariablen

Ein Nachteil der im letzten Abschnitt behandelten Verfahren besteht darin, dass ggf. lediglich einzelne Parameter für sich geshrinkt werden. Unter Verwendung

der Ridge-Penalty ist auch keine Parameterselektion möglich und durch das Verfahren von Friedman et al. (2010) werden ggf. einzelne Parameter aus dem Modell genommen. Dies bedeutet, dass die entsprechenden Koeffizienten unabhängig von denjenigen, die zwar zur selben Variable, jedoch zu einer anderen Responsekategorie gehören, ggf. auf Null gesetzt werden, wodurch keine direkte Variablenselektion stattfindet. Um dieser Problematik entgegenzuwirken, soll, entsprechend dem Lasso-Verfahren (vgl. Tibshirani, 1996), ein Lasso-Penalisierungsterm Anwendung finden, welcher bei einer symmetrischen Nebenbedingung als

$$J(\boldsymbol{\beta}) = \sum_{j=1}^{p} (\beta_{1j}^2 + \ldots + \beta_{Kj}^2)^{1/2} \tag{21}$$

und bei Verwendung der K-ten Responsekategorie als Referenzkategorie als

$$J(\boldsymbol{\beta}) = \sum_{j=1}^{p} (\beta_{1j}^2 + \ldots + \beta_{qj}^2)^{1/2} \tag{22}$$

definiert wird (vgl. bzgl. dieser Penalisierungsterme Tutz, 2011). Die Umsetzung von letzterer Darstellung wird in Kapitel 4 behandelt.

4 Algorithmus zur Merkmalsauswahl

Der im Kapitel 4.4 vorgestellte Algorithmus baut auf das Paper von Meier et al. (2008) bzw. ein darin vorgestelltes Verfahren, bei welchem von binärem Response und kategorialen sowie metrischen Einflussgrößen ausgegangen wird, auf. Die entsprechende Methodik wird in Abschnitt 4.2 behandelt.

4.1 Lasso

Zunächst sollen die Grundlagen für die in den Kapiteln 4.3 und 4.4 gezeigten Algorithmen gelegt werden. Ausgangspunkt ist der Lasso. Die folgenden Ausführungen beziehen sich auf das Paper von Tibshirani (1996).

Der Lasso-Schätzer maximiert die allgemeine Log-Likelihood $l(\boldsymbol{\beta})$ unter der Nebenbedingung $\sum_{j=1}^{p} \beta_j \leq t$ $(t \geq 0)$, was gleichbedeutend mit der Darstellung

$$\arg\max_{\boldsymbol{\beta}} \, l(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1, \quad \text{mit} \quad \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|, \tag{23}$$

für einen entsprechenden $\boldsymbol{\beta}$-Vektor ist. Durch dieses Vorgehen werden die Koeffizienten geshrinkt und ggf. auf Null gesetzt. Dieses Verhalten ist naheliegenderweise von λ abhängig. Je größer der entsprechende Wert ist, desto „kleiner“, in dem Sinne, dass tendenziell mehr Parameter auf Null geschätzt werden bzw. die Parameter kleinere Schätzwerte erhalten, werden die resultierenden Modelle. Für jeden Koeffizienten ergeben sich somit über das λ hinweg Pfade, ausgehend von einem $\lambda_{\max}$, bei welchem bspw. alle Koeffizienten auf Null geschätzt werden, bis hin zu einem $\lambda_{\min}$. Setzt man $\lambda_{\min} = 0$ erhält man an dieser Stelle klarerweise den ML-Schätzer. Aus diesen Pfaden wird dann über ein Kriterium, z.B. generalisierte Kreuzvalidierung (vgl. Tibshirani, 1996, S. 276) das „optimale“ λ ausgewählt. Innerhalb dieser Pfade ist es dabei möglich, dass die Vorzeichen der Koeffizientenschätzer wechseln.

Für Details zum Lasso-Verfahren sei hier auf Tibshirani (1996) verwiesen. Dort werden zudem Simulationen im linearen Modell gezeigt (vgl. Tibshirani, 1996, S. 279 ff.).

4.2 Group Lasso

Eine Erweiterung der Idee des Lasso wurde durch das „Group Lasso“ (vgl. Yuan and Lin, 2006) eingeführt. Hierbei ist es, im Gegensatz zum einfachen Lasso, möglich, gesamte Prädiktorgruppen, wie sie bspw. bei kategorialen Einflussgrößen auftreten, zu regularisieren. Zur Veranschaulichung wird, wie im Paper von Meier et al. (2008, S. 54 ff.), worauf sich auch die folgenden Ausführungen und die Darstellungen in Kapitel 4.3 beziehen, das binäre logistische Modell betrachtet. Dabei wird angenommen, dass sich insgesamt n Beobachtungen mit jeweils G

Prädiktoren im Modell befinden und entsprechend $y_i \in \{0, 1\}$ $(i = 1, \ldots, n)$ gilt. Zudem können die G Einflussgrößen sowohl kategorial als auch metrisch sein, was durch df_g $(g = 1, \ldots, G)$ gekennzeichnet wird. Für metrische Kovariablen gilt, wie in Abschnitt 2.1 bereits dargestellt wurde, $\mathrm{df}_g = 1$ und für kategoriale $\mathrm{df}_g = p_g - 1$ $(g = 1, \ldots, G)$, wobei mit p_g die Anzahl der Kategorien des entsprechenden Prädiktors bezeichnet wird. Zu schätzen sind $p + 1$ Koeffizienten, mit $p = \sum_{g=1}^{G} \mathrm{df}_g$. Für die zu minimierende Zielfunktion ergibt sich damit

$$S_\lambda(\boldsymbol{\beta}) = -l(\boldsymbol{\beta}) + \lambda \sum_{g=1}^{G} s(\mathrm{df}_g) \|\boldsymbol{\beta}_{.g}\|_2, \tag{24}$$

wobei $s(\cdot)$ eine Funktion der Freiheitsgrade df_g $(g = 1, \ldots, G)$ sei und für die negative Log-Likelihood $(-l(\boldsymbol{\beta}))$ Gleichung (4) aus Kapitel 2.1 verwendet wird. Dieser Strafterm stellt nun sicher, dass ein Prädiktor entweder insgesamt, oder gar nicht in das Modell aufgenommen wird und kann als Mischung aus Ridge-Regression und Lasso betrachtet werden (vgl. Meier et al., 2008, S. 55). Die Minimierung der Funktion $S_\lambda(\cdot)$ aus Gleichung (24) kann nun anhand des folgenden, im Paper von Meier et al. (2008, S. 55 ff.) vorgestellten Algorithmus geschehen.

4.3 Algorithmus für das binäre logistische Modell

Von nun an sei $\boldsymbol{X}$ die Designmatrix

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \ldots & x_{1p} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \ldots & x_{np} \end{pmatrix}.$$

Man nehme als $\boldsymbol{X}_g$ eine Matrix der Dimension $n \times \mathrm{df}_g$ $(g = 1, \ldots, G)$, welche aus der/den entsprechenden Spalte(n) der Designmatrix besteht, an. Hierbei sei außerdem angenommen, dass $\boldsymbol{X}_g$ $(g = 1, \ldots, G)$ vollen Rang besitzt und somit eine blockweise Orthonormalisierung möglich ist, welche als Vorbereitung für den Algorithmus durchgeführt wird. Es gilt somit $\boldsymbol{X}_g^T \boldsymbol{X}_g = \boldsymbol{I}_{\mathrm{df}_g}$ $(g = 1, \ldots, G)$, mit $\boldsymbol{I}_{\mathrm{df}_g}$ als Identitätsmatrix der Dimension $\mathrm{df}_g \times \mathrm{df}_g$. Im Paper von Meier et al. (2008) wird eine leicht modifizierte Version verwendet, sodass $\boldsymbol{X}_g^T \boldsymbol{X}_g = n I_{\mathrm{df}_g}$ $(g = 1, \ldots, G)$ gilt (vgl. Meier et al., 2008, S. 55).

Im eigentlichen Algorithmus wird nun die Minimierung von $S_\lambda(\cdot)$ kovariablenweise durchgeführt. Während ein $\boldsymbol{\beta}_{.g}$ $(g = 1, \ldots, G)$ angepasst wird, werden alle anderen bei einem Wert festgehalten. Insgesamt wird folgendes Verfahren angewendet (vgl. Meier et al., 2008, S. 56):

1. Wähle für $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ einen Startvektor.

2. Berechne den Intercept aus $\arg\min_{\beta_0}\{S_\lambda(\boldsymbol{\beta})\}$.

3. Für einen Durchlauf durch $g = 1, \ldots, G$ gehe folgendermaßen vor:
   - Falls $\|\boldsymbol{X}_g^T(\boldsymbol{y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_2 \leq \lambda s(\mathrm{df}_g)$, setze $\boldsymbol{\beta}_{.g} = 0$
   - ansonsten: Bestimme $\boldsymbol{\beta}_{.g}$ aus $\arg\min_{\boldsymbol{\beta}_{.g}}\{S_\lambda(\boldsymbol{\beta})\}$.
4. Wiederhole die Schritte 2 und 3 bis zur Konvergenz.

Dabei bezeichnet $\boldsymbol{\beta}_{-g}$ den Vektor $\boldsymbol{\beta}$, wobei der Vektor $\boldsymbol{\beta}_g$ bzw., falls $\mathrm{df}_g = 1$ gilt, der Wert von β_g auf Null gesetzt wird. $\boldsymbol{y}$ steht wiederum für den Responsevektor und $\boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}}$ für den geschätzten Wahrscheinlichkeitsvektor der Kategorienzugehörigkeit ohne Berücksichtigung des Vektors $\boldsymbol{\beta}_g$ bzw. des Wertes von β_g. Zudem wird $s(\mathrm{df}_g) = df_g^{1/2}$ gesetzt. Im dritten Schritt dieses Algorithmus findet die Variablenselektion statt. Hierfür wird zunächst überprüft, ob die Variable überhaupt angepasst werden soll.

Durch den Term $\|\boldsymbol{X}^T\boldsymbol{y}\|_2$ bzw. $\|\boldsymbol{X}_g^T\boldsymbol{y}\|_2$ wird die Korrelation zwischen $\boldsymbol{X}$ und $\boldsymbol{y}$ bzw. $\boldsymbol{X}_g$ und $\boldsymbol{y}$ $(g = 1, \ldots, G)$, da $\boldsymbol{X}_g$ $(g = 1, \ldots, G)$ blockweise orthonormalisiert wurde, beschrieben. Das Addieren von $\boldsymbol{y}$ mit $-\boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}}$ stellt sicher, dass der bereits erklärte Anteil der Wahrscheinlichkeiten für $\boldsymbol{y}$ ohne Berücksichtigung des g-ten Prädiktors miteinbezogen wird, wodurch sich insgesamt aus der Norm ein Maß für die „Wichtigkeit“ des g-ten Prädiktors ergibt.

Im Paper von Meier et al. (2008) wird noch auf Folgendes hingewiesen: Im dritten Schritt wird zunächst überprüft, ob sich das Minimum an der nicht differenzierbaren Stelle 0 befindet. Falls dies nicht der Fall ist, so kann ein Standardverfahren verwendet werden und die Werte aus der letzten Iteration können als Startwerte verwendet werden. Falls die Variable im vorhergenden Durchgang nicht in das Modell aufgenommen wurde, so wird erst ein kleiner Schritt in die entgegengesetzte Richtung des Gradienten der negativen Log-Likelihood durchgeführt, um sicherzustellen, dass an einer differenzierbaren Stelle gestartet wird (vgl. Meier et al., 2008, S. 56).

Über die Verwendung einer λ-Sequenz ergeben sich für die β-Werte Pfade (vgl. Meier et al., 2008, S. 56 für ein Beispiel für entsprechende Pfade). Im Paper von Meier et al (2008, S. 57 ff.) ist noch eine weitere Methodik, welche ebenfalls für das oben beschriebene penalisierte Schätzproblem bzgl. des binären Modells angewendet werden kann, beschrieben. Diese ist über das Paket „`grplasso`“ (Meier, 2009) für die Software „`R`“ (R Development Core Team, 2010) abrufbar (vgl. Meier et al., 2008, S. 59). Bzgl. der entsprechenden Theorie sei an dieser Stelle auf das Paper von Meier et al. (2008, S. 57 ff.) verwiesen.

4.4 Algorithmus für das multinomiale logistische Modell

Der im letzten Abschnitt vorgestellte Algorithmus soll nun dahingehend modifiziert werden, als dass er zur Merkmalsauswahl im multinomialen Modell verwendet werden kann. Man beachte, dass im folgenden Kapitel 4.4.1 ausschließlich von Kovariablen von metrischem Skalenniveau ausgegangen wird.

4.4.1 Metrische Kovariablen

Die zu minimierende Zielfunktion ist hierbei fast dieselbe wie Funktion (24):

$$\begin{aligned}\widetilde{S}_\lambda(\boldsymbol{\beta}) &= -l(\boldsymbol{\beta}) + s(q)\lambda \sum_{j=1}^{p} \|\boldsymbol{\beta}_{.j}\|_2 \\ &= -l(\boldsymbol{\beta}) + s(q)\lambda \sum_{j=1}^{p} (\beta_{1j}^2 + \ldots + \beta_{qj}^2)^{1/2},\end{aligned} \tag{25}$$

wobei $l(\boldsymbol{\beta})$ nun für die Log-Likelihood aus dem multinomialen logistischen Modell (Gleichung (6) aus Kapitel 2.2.1) steht und die K-te Responsekategorie als Referenzkategorie verwendet wird. Der Unterschied zur Gleichung (24) besteht darin, dass der Term $s(\mathrm{df}_g)$ durch $s(q)$ ersetzt wurde und, neben der ausgetauschten Log-Likelihood, $\boldsymbol{\beta}$ entsprechend dem multinomialen Modell angepasst ist. Die Kategorien einer Kovariable aus dem vorherigen Abschnitt zum Group Lasso beim binären Logitmodell entsprechen nun den Kategorien des Response, wodurch die vorherigen $\boldsymbol{\beta}_{.g}$-Vektoren ($g = 1, \ldots, G$) durch $\boldsymbol{\beta}_{.j}$ ($j = 1, \ldots, p$) ersetzt werden und somit alle dieselbe Länge besitzen. Daher kann auch der Term $s(\mathrm{df}_g)$, welcher eine entsprechende Gewichtung der $\boldsymbol{\beta}_{.g}$ ($g = 1, \ldots, G$) sicherstellte, gegen $s(q) = q^{1/2}$, was von g ($g = 1, \ldots, G$) unabhängig ist und somit vor die Summe geschrieben werden darf, ausgetauscht werden. Würde man ausschließlich globale Einflussgrößen verwenden, könnte dieser Term ganz weggelassen werden, da er bereits von λ abgedeckt wäre. In Kapitel 4.4.3 wird dargestellt, warum er für eine allgemeine Anwendung des Algorithmus notwendig ist.

Für eine alternative Darstellung der Log-Likelihood wird die Designmatrix $\boldsymbol{X}$

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \ldots & x_{1p} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \ldots & x_{np} \end{pmatrix}$$

und eine Umformulierung der Vektoren $\boldsymbol{\beta}$ und $\boldsymbol{y}$ zu

$$\boldsymbol{\beta}^\star = \begin{pmatrix} \beta_{10} & \beta_{11} & \ldots & \beta_{1p} \\ \vdots & \vdots & & \vdots \\ \beta_{q0} & \beta_{q1} & \ldots & \beta_{qp} \end{pmatrix}$$

bzw.

$$\boldsymbol{Y} = \begin{pmatrix} \boldsymbol{y}_1^T & 1 - y_{11} - \ldots - y_{1q} \\ \vdots & \vdots \\ \boldsymbol{y}_n^T & 1 - y_{n1} - \ldots - y_{nq} \end{pmatrix}$$

durchgeführt. Analog dazu wird

$$\boldsymbol{\pi} = \begin{pmatrix} \boldsymbol{\pi}_1^T & 1 - \pi_{11} - \ldots - \pi_{1q} \\ \vdots & \vdots \\ \boldsymbol{\pi}_n^T & 1 - \pi_{n1} - \ldots - \pi_{nq} \end{pmatrix}$$

definiert. Hierbei wird ebenfalls für $\boldsymbol{X}$ eine blockweise Orthonormalisierung durchgeführt, sodass $\boldsymbol{x}_j^T \boldsymbol{x} = 1$ $(j = 1, \ldots, p)$ gilt. Die Log-Likelihood kann in dieser Notation über

$$l(\boldsymbol{\beta}) = \sum \operatorname{diag}(\boldsymbol{X}\tilde{\boldsymbol{\beta}}^T \boldsymbol{Y}^T) - \sum_{i=1}^{n} \log(1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T \boldsymbol{\beta}_{s.})), \tag{26}$$

wobei über die Funktion $\operatorname{diag}(\cdot)$ ein Vektor mit den Diagonalelementen der übergebenen Matrix als Argument gewonnen wird und mit dem Summenzeichen ohne Indices die Summe über die Einträge des darauffolgenden Vektors gemeint ist, dargestellt werden. $\tilde{\boldsymbol{\beta}}$ beschreibt die Matrix $(\boldsymbol{\beta}^{\star T}, \boldsymbol{0})^T$, bei welcher also ein Nullvektor als $q+1$-te Zeile eingefügt wurde. Wird beim Response als Referenzkategorie eine andere Kategorie als die letzte verwendet, so muss dieser Nullvektor in die entsprechende Zeile gesetzt werden.

Der Vorteil dieser Formulierung ist, dass nun relativ einfach bzgl. eines Koeffizientenvektors $\boldsymbol{\beta}_{.j}$ zu einer Kovariable j $(j = 1, \ldots, p)$ optimiert werden kann. Bei festgehaltenem $\boldsymbol{\beta}_{.-j}$ ist die penalisierte Log-Likelihood, also die Zielfunktion $\widetilde{S}(\boldsymbol{\beta}_{.j})$ in Abhängigkeit von $\boldsymbol{\beta}_{.j}$ nun formulierbar als

$$\begin{aligned}
\widetilde{S}(\boldsymbol{\beta}_{.j}) = & \sum \operatorname{diag}(\boldsymbol{X}\tilde{\boldsymbol{\beta}}_{.-j}^T \boldsymbol{Y}^T) + \sum \operatorname{diag}(\boldsymbol{X}\tilde{\boldsymbol{\beta}}_{.j}^T \boldsymbol{Y}^T) \\
& - \sum_{i=1}^{n} \log(1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_{i,-j}^T \boldsymbol{\beta}_{s,-j} + x_{ij}^T \beta_{sj})) \\
& - s(q)\lambda \sum_{j^\star \neq j} (\beta_{1j^\star}^2 + \ldots + \beta_{qj^\star}^2)^{1/2} - s(q)\lambda(\beta_{1j}^2 + \ldots + \beta_{qj}^2)^{1/2}, \\
& j = 1, \ldots, p.
\end{aligned} \tag{27}$$

Das Komma bei $\boldsymbol{x}_{i,-j}$ und $\boldsymbol{\beta}_{s,-j}$ wurde lediglich zur Erleichterung der Lesbarkeit eingefügt und hat inhaltlich keinerlei Bedeutung. Gemeint ist also die i-te Beobachtung ohne Berücksichtigung der j-ten Einflussgröße bzw. die s-te Responsekategorie ohne Berücksichtigung der j-ten Einflussgröße.

Analog zu oben durchläuft der für das multinomiale Logitmodell angepasste Algorithmus nun folgende Schritte:

1. Bestimme einen Startvektor für $\boldsymbol{\beta}$.
2. Berechne den Interceptvektor $\boldsymbol{\beta}_{.0}$ als $\arg\min_{\boldsymbol{\beta}_{.0}}\{\widetilde{S}_\lambda(\boldsymbol{\beta})\}$.
3. Für einen Durchlauf durch $j = 1, \ldots, p$ gehe folgendermaßen vor:
   - Falls $\|\boldsymbol{x}_j^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{.-j}})\|_2 \leq s(q)\lambda$, setze $\boldsymbol{\beta}_{.j} = \boldsymbol{0}$.
   - Bestimme ansonsten $\boldsymbol{\beta}_{.j}$ über $\arg\min_{\boldsymbol{\beta}_{.j}}\{\widetilde{S}_\lambda(\boldsymbol{\beta})\}$.
4. Wiederhole die Schritte 2 und 3 bis zur Konvergenz.

$\boldsymbol{\beta}_{\cdot j}$ $(j = 1, \ldots, p)$ beschreibt den Koeffizientenvektor einer Variable über alle Kategorien des Response, mit Ausnahme der Referenzkategorie, und kann aus dem Gesamtvektor $\boldsymbol{\beta}$ (vgl. Darstellung (7) aus Abschnitt 2.2) bzw. aus $\boldsymbol{\beta}^{\star}$ gewonnen werden. Im binären Logitmodell würde sich für diesen Vektor ein Skalar ergeben. $\boldsymbol{x}_j$ $(j = 1, \ldots, p)$ steht für die j-te Spalte $(j = 1, \ldots, p)$ aus der Designmatrix. Es sei nochmals darauf hingewiesen, dass in diesem Kapitel 4.4.1 ausschließlich von metrischen Einflussgrößen ausgegangen wird und somit in der Schreibweise aus dem letzten Abschnitt $G = p$ gilt. $\boldsymbol{Y}$ ist die oben eingeführte Responsematrix und $\boldsymbol{\pi}_{\boldsymbol{\beta}_{\cdot -j}}$ steht wiederum für die Wahrscheinlichkeitsmatrix ohne die Berücksichtigung von $\boldsymbol{\beta}_{\cdot j}$.

Für die einzelnen β-Werte ergeben sich, analog zum Verfahren von Meier et al. (2008), für eine λ-Sequenz Pfade, welche jedoch nun entweder nach Responsekategorien oder nach Einflussgrößen aufgeteilt werden können, um sie grafisch darzustellen. Eine solche Veranschaulichung ist im Kapitel 4.6, in welchem der Algorithmus etwas modifiziert wird, zu finden.

Um ein konkretes Modell zu erhalten, muss dann selbstverständlich ein λ-Wert ausgewählt werden, was im Kapitel 5 („Anwendung des Algorithmus“) über eine fünffache Kreuzvalidierung mit dem prädiktiven „Brier Score“ (Brier, 1950) als Kriterium geschieht. Der Brier Score wird hier als

$$\text{Brier} = \sum_{i=1}^{n} (\boldsymbol{y}_i - \hat{\boldsymbol{\pi}}_i)^2, \tag{28}$$

wobei $\boldsymbol{y}_i$ die i-te Zeile von $\boldsymbol{Y}$ und $\boldsymbol{\pi}_i$ die i-te Zeile von $\boldsymbol{\pi}$, dem geschätzten $\boldsymbol{\pi}$ beschreibt. Das $\boldsymbol{Y}$ ist hierbei aus dem Testdatensatz, während $\boldsymbol{\pi}$ aus dem jeweiligen Trainingsdatensatz gewonnen wurde.

Neben dieser Vorgehensweise wären natürlich auch andere Modellwahlkriterien, wie z.B. die Devianz (vgl. Fahrmeir et al., 2007, S. 205) in Verbindung mit einer Kreuzvalidierung, anwendbar. Eine Funktion, welche eine fünffache Kreuzvalidierung bzgl. des prädiktiven Brier Score und bzgl. der prädiktiven Log-Likelihood (= prädiktive Devianz / 2) durchführt, ist auf der beigefügten CD zu finden. Auch wenn dies in der Theorie nicht von Belang ist, sei hier noch erwähnt, dass, falls mindestens zwei verschiedene λ-Werte zum gleichen Wert bzgl. des Brier Scores bzw. der prädiktiven Log-Likelihood führen, der höchste als der optimale verwendet wird. Es wird hiermit lediglich die Eindeutigkeit des entsprechenden Rückgabewertes der Funktion „`plregr.cv`“ (s. die dieser Arbeit beigefügte CD) gesichert.

Bei einer k-fachen Kreuzvalidierung wird der Datensatz in k Teile, welche in etwa gleich groß sind, unterteilt. Anschließend dient in jedem der k Durchgänge einer der Subdatensätze als Test- und die restlichen als Trainingsdatensatz, sodass jeder dieser Teildatensätze einmal als Testdatensatz diente. Anhand der Trainingsdaten werden die Parameter geschätzt und über die Testdaten validiert. Dies wird für jeden λ-Wert durchgeführt. Es wird anschließend derjenige λ-Wert

verwendet, welcher den niedrigsten Brier Score zur Folge hat (vgl. Fahrmeir et al., 2007, S. 162, für eine sehr ähnliche Darstellung der Kreuzvalidierung). In der Funktion `plregr.cv`, welche bei der Durchführung einer Kreuzvalidierung für die in Kapitel 5 gezeigten Analysen Verwendung fand, geschieht die Zuteilung der Beobachtungen in eine der k Gruppen zufällig.

Das Verfahren wird nun um zwei weitere Arten von Kovariablen erweitert. Damit ist es zum einen auf kategoriale Prädiktoren von nominalem Skalenniveau und auf kategorienspezifische Variablen mit metrischem Skalenniveau anwendbar.

4.4.2 Kategoriale Kovariablen

Zunächst sei nochmals auf die allgemeine Schreibweise hingewiesen. Es wird angenommen, dass für ein Modell G Kovariablen, welche entweder von nominalem oder metrischem Skalenniveau sind, zur Verfügung stehen. Einige hiervon werden im Endmodell ggf. nicht berücksichtigt. Ist eine Einflussgröße g ($g = 1, \ldots, G$) metrisch, so kann diese Kovariable unverändert in die Designmatrix aufgenommen werden. Entsprechend gilt für diese Einflussgröße $\mathrm{df}_g = 1$. Ist eine Kovariable hingegen kategorial (mit mindestens zwei Katogorien), so wird diese in df_g Effekt-kodierte Variablen mit $\mathrm{df}_g = (\text{Anzahl der Kategorien} - 1)$ umkodiert. Es gilt $p = \sum_{g=1}^{G} \mathrm{df}_g$. Beim Übergang zu kategorialen Kovariablen erhält man somit, falls $\mathrm{df}_g > 1$, die Matrix $\boldsymbol{X}_g$ ($g = 1, \ldots, G$) mit den df_g zu g gehörenden Effekt-kodierten Variablen, wodurch sich für $\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})$ ebenfalls eine Matrix ergibt. Diese besitzt, je nach Anzahl der Kategorien der Variable g, df_g Zeilen ($g = 1, \ldots, G$). Damit wird eine entsprechende Norm benötigt, wofür in diesem Algorithmus die „Frobenius-Norm“ (vgl. bspw. Schaback und Wendlang, 2005, S. 53), welche für ein $\boldsymbol{X}$ der Dimension $n \times p$ mit den Einträgen x_{ij} ($i = 1, \ldots, n$, $j = 1, \ldots, p$) definiert ist als

$$\|\boldsymbol{X}\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{p} x_{ij}^2 \right)^{1/2}, \tag{29}$$

verwendet wird. Gilt $\mathrm{df}_g = 1$, so ist $\boldsymbol{x}_g$ ($g = 1, \ldots, G$) ein Vektor und die L2-Norm ist, wie im für metrische Kovariablen vorgestellten Algorithmus, anwendbar. Eine blockweise Orthonormalisierung wird auch hier durchgeführt, sodass $\boldsymbol{X}_g^T \boldsymbol{X} = I_g$ bzw. $\boldsymbol{x}_g^T \boldsymbol{x}_g = 1$ ($g = 1, \ldots, G$) gilt. Aus der Abhängigkeit der Dimension der Matrix $\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})$ wird zudem direkt ersichtlich, dass hier eine entsprechende Gewichtung über $s(\mathrm{df}_g) = \mathrm{df}_g^{1/2}$ ($g = 1, \ldots, G$) in einer zum im Abschnitt 4.2 vorgestellten Algorithmus analogen Weise stattfinden muss. Die zu optimierende Funktion $\widetilde{S}(\boldsymbol{\beta})$ lässt sich in einer allgemeinen Form als

$$\widetilde{S}_\lambda(\boldsymbol{\beta}) = -l(\boldsymbol{\beta}) + s(q)\lambda \sum_{g=1}^{G} s(\mathrm{df}_g) \|\boldsymbol{\beta}_{.g}\|_F \tag{30}$$

formulieren. Diese ist für metrische bzw. binäre und kategoriale Kovariablen anwendbar.

Eine zur Gleichung 26 bzw. 27 äquivalente Formulierung von $l(\boldsymbol{\beta})$ bzw. $\widetilde{S}(\boldsymbol{\beta}_{.g})$ ist ebenfalls möglich. Jedoch ist diese mehr aus programmiertechnischer Sicht interessant und nur sehr umständlich explizit darstellbar. Somit sei hier lediglich auf die Funktion „`likeli.j`“, welche innerhalb der Funktion `plregr` verwendet wird, verwiesen. In der Funktion `plregr` ist der in Kapitel 4.6 vorgestellte Algorithmus umgesetzt.

Bei globalen Einflussgrößen wird folgende Vorgehensweise verwendet:

1. Bestimme einen Startvektor für $\boldsymbol{\beta}$.
2. Berechne den Interceptvektor $\boldsymbol{\beta}_{.0}$ als $\arg\min_{\boldsymbol{\beta}_{.0}}\{\widetilde{S}_\lambda(\boldsymbol{\beta})\}$.
3. Für einen Durchlauf durch $g = 1, \ldots, G$ gehe folgendermaßen vor:
   - Falls $\|\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_F \leq s(q)s(\mathrm{df}_g)\lambda$, setze $\boldsymbol{\beta}_{.g} = \mathbf{0}$.
   - Bestimme ansonsten $\boldsymbol{\beta}_{.g}$ über $\arg\min_{\boldsymbol{\beta}_{.g}}\{\widetilde{S}_\lambda(\boldsymbol{\beta})\}$.
4. Wiederhole die Schritte 2 und 3 bis zur Konvergenz.

Hierbei wurde die Unterscheidung zwischen $\boldsymbol{X}_g$ und $\boldsymbol{x}_g$, also ob $\mathrm{df}_g > 1$ ($g = 1, \ldots, G$) ist oder nicht, vernachlässigt, um eine einheitliche Darstellung zu erhalten. In dieser Form ist der Algorithmus nun allgemein für metrische und kategoriale Kovariablen anwendbar.

4.4.3 Kategorienspezifische Kovariablen

Auf kategorienspezifische Kovariablen ist der Algorithmus ebenfalls erweiterbar, wofür zunächst noch einige anschließend verwendete Schreibweisen eingeführt werden. Es wird außerdem angenommen, dass $L > 2$ und $K > 2$ gilt. Für $L \leq 2$ und/oder $K = 2$ ergeben sich nur geringfügige Umformulierungen, welche aus den hier gezeigten Darstellungen direkt ableitbar sind.

Im Folgenden wird mit $\boldsymbol{w}_{ir.}$ ($r = 1, \ldots, q$) der Vektor bezeichnet, von welchem bereits der zur Referenzkategorie gehörende Vektor $\boldsymbol{w}_{iK}$ subtrahiert wurde. Es gilt $\boldsymbol{w}_{ir.} = (w_{ir1}, \ldots, w_{irL})$, wodurch sowohl nach Beobachtung i ($i = 1, \ldots, n$), als auch nach Responsekatogorie r ($r = 1, \ldots, q$) oder kategorienspezifischer Einflussgröße l ($l = 1, \ldots, L$) aufgespaltet werden kann. Hiermit werden die Matrizen

$$\boldsymbol{W}_{i..} = \begin{pmatrix} \boldsymbol{w}_{i1.}^T \\ \vdots \\ \boldsymbol{w}_{iq.}^T \end{pmatrix}, \quad i = 1, \ldots, n,$$

und daraus wiederum die Matrix

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{W}_{1..} \\ \vdots \\ \boldsymbol{W}_{n..} \end{pmatrix}$$

gebildet. Die Matrix $\boldsymbol{W}$ ist damit von der Dimension $(n \cdot q) \times L$ und wird für die weitere Anwendung, entsprechend zu $\boldsymbol{X}$, blockweise orthonormalisiert, sodass $\boldsymbol{w}_{..l}^T \boldsymbol{w}_{..l} = 1$, mit $\boldsymbol{w}_{..l} = (\boldsymbol{w}_{1.l}, \ldots, \boldsymbol{w}_{n.l})$ $(l = 1, \ldots, L)$ gilt. Zur Vereinfachung der Schreibweise wird $\boldsymbol{w}_{..l}$ im Folgenden mit $\boldsymbol{w}_l$ bezeichnet.

Die ursprüngliche Zielfunktion $\widetilde{S}_\lambda(\boldsymbol{\beta})$ muss entsprechend zu

$$\begin{aligned} \widetilde{S}_\lambda(\boldsymbol{\beta}, \boldsymbol{\gamma}) &= -l(\boldsymbol{\beta}, \boldsymbol{\gamma}) + s(q)\lambda \sum_{j=1}^{p} \|\boldsymbol{\beta}_{.j}\|_2 + \lambda \|\boldsymbol{\gamma}\|_1 \qquad (31) \\ &= -l(\boldsymbol{\beta}, \boldsymbol{\gamma}) + s(q)\lambda \sum_{j=1}^{p} (\beta_{1j}^2 + \ldots + \beta_{qj}^2)^{1/2} + \lambda \sum_{l=1}^{L} |\gamma_l|, \end{aligned}$$

mit $l(\boldsymbol{\beta}, \boldsymbol{\gamma})$ aus Kapitel 2.2.2 (Gleichung (11)), umformuliert werden.

Anstatt der Matrix $\boldsymbol{X}_g$ bzw. dem Vektor $\boldsymbol{x}_g$ $(g = 1, \ldots, G)$ wird nun entsprechend die l-te Spalte von $\boldsymbol{W}$ verwendet und $\boldsymbol{\pi}_{\boldsymbol{\gamma}_{-l}}$, mit $\boldsymbol{\pi}_{\boldsymbol{\gamma}_{-l}}$ als $\boldsymbol{\pi}$ ohne die Berücksichtigung von γ_l $(l = 1, \ldots, L)$, jedoch unter Verwendug des gesamten $\boldsymbol{\beta}$-Vektors, berechnet. Hierbei ist zu beachten, dass mit $\boldsymbol{w}_l$ bereits der mit $-\boldsymbol{w}_{lK}$ addierte Vektor bezeichnet wird. Wie in Gleichung (31) bereits dargestellt, wird für den globalen Koeffizientenvektor γ keine Gewichtung bzw. das Gewicht 1 und somit der „gewöhnliche“ Lasso-Strafterm verwendet. Dies hängt selbstverständlich damit zusammen, dass pro Einflussgröße lediglich ein Koeffizient anzupassen ist. Wie in Kapitel 4.4.1 bereits erwähnt, wird erst durch die Berücksichtigung von kategorienspezifischen Einflussgrößen eine Gewichtung des Strafterms mit $s(q)$ für die kategorienspezifischen Koeffizienten nötig.

Eine weitere Problematik macht sich schließlich bei der praktischen Umsetzung bemerkbar, da $\boldsymbol{w}_l$ $(l = 1, \ldots, L)$ nur dann dieselbe Länge, wie der Vektor $\boldsymbol{x}_g$ $(g = 1, \ldots, G)$, besitzt, falls die Responsevariable binär ist, wobei die Variable g entweder ebenfalls nur zwei Kategorien besitzt oder von metrischem Skalenniveau ist. I.A. hat ein solcher Vektor $\boldsymbol{w}_l$ $(l = 1, \ldots, L)$ die Länge $n \cdot q$ und kann nicht mit der Matrix $\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\gamma}_{-l}}$ $(l = 1, \ldots, L)$ multipliziert werden, ohne vorher entsprechend umformatiert zu werden. Zudem ist die Berechnung von $\boldsymbol{\pi}_{-l}$ $(l = 1, \ldots, L)$ nicht so direkt, wie bei globalen Einflussgrößen, zumindest in der für den Algorithmus verwendeten Formulierung, durchführbar. Beides ist jedoch für die Durchführung des hier vorgestellten Verfahrens notwendig.

Zunächst werden daher die Vektoren

$$\boldsymbol{w}_i^\star = \boldsymbol{W}_{i..}\boldsymbol{\gamma} = \begin{pmatrix} \boldsymbol{w}_{i1.}^T \\ \vdots \\ \boldsymbol{w}_{iq.}^T \end{pmatrix} \boldsymbol{\gamma} = \begin{pmatrix} \boldsymbol{w}_{i1.}^T \boldsymbol{\gamma} \\ \vdots \\ \boldsymbol{w}_{iq.}^T \boldsymbol{\gamma} \end{pmatrix}, \quad i = 1, \ldots, n,$$

und zur Veranschaulichung

$$\boldsymbol{w}^{\star} = \begin{pmatrix} \boldsymbol{w}_1^{\star} \\ \vdots \\ \boldsymbol{w}_n^{\star} \end{pmatrix}$$

der Länge q bzw. $n \cdot q$, gebildet. Daraus wird wiederum eine Matrix

$$\boldsymbol{W}^{\star} = \begin{pmatrix} \boldsymbol{w}_1^{\star^T} \\ \vdots \\ \boldsymbol{w}_n^{\star^T} \end{pmatrix} = \begin{pmatrix} \boldsymbol{w}_{11.}^T\boldsymbol{\gamma} & \dots & \boldsymbol{w}_{1q.}^T\boldsymbol{\gamma} \\ \vdots & & \vdots \\ \boldsymbol{w}_{n1.}^T\boldsymbol{\gamma} & \dots & \boldsymbol{w}_{nq.}^T\boldsymbol{\gamma} \end{pmatrix},$$

der Dimension $n \times q$ konstruiert, in welcher alle zu einer Beobachtung i ($i = 1, \dots, n$) gehörenden Ausprägungen nebeneinander in eine Zeile geschrieben werden. Die Matrix $\boldsymbol{W}^{\star^T}$ kann nun mit der Matrix $\boldsymbol{Y} - \boldsymbol{\pi}$ multipliziert werden. Dies ist für die Durchführung des dritten Schrittes des Algorithmus notwendig. Dort ist es jedoch entscheidend, dass die l-te Variable ($l = 1, \dots, L$) gesondert berücksichtigt wird. Dies wird dadurch sichergestellt, dass zunächst der Vektor

$$\boldsymbol{w}_{-l}^{\star} = \boldsymbol{W}_{-l}\boldsymbol{\gamma}_{-l}$$

konstruiert wird, wobei $\boldsymbol{W}_{-l}$ die Matrix $\boldsymbol{W}$ ohne die l-te Spalte und $\boldsymbol{\gamma}_{-l}$ entsprechend den Vektor $\boldsymbol{\gamma}$ ohne den l-ten Eintrag bezeichnet. Aus dem Vektor $\boldsymbol{w}_{-l}^{\star}$ wird dann analog zu $\boldsymbol{W}^{\star}$ die Matrix

$$\boldsymbol{W}_{-l}^{\star} = \begin{pmatrix} \boldsymbol{w}_{11,-l}^T\gamma_{-l} & \dots & \boldsymbol{w}_{1q,-l}^T\gamma_{-l} \\ \vdots & & \vdots \\ \boldsymbol{w}_{n1,-l}\gamma_{-l} & \dots & \boldsymbol{w}_{nq,-l}^T\gamma_{-l} \end{pmatrix},$$

in welcher die l-te Einflussgröße nicht berücksichtigt wird, gebildet. Verwendet man die Matrix $\boldsymbol{W}_{-l}^{\star}$ nun für die Berechnung von $\boldsymbol{\eta}$ aus Kapitel 2.2.2 (Gleichung (10)), so kann auf diese Weise ein $\boldsymbol{\pi}_{-l}$ ($l = 1, \dots, L$) berechnet werden.

Zusätzlich wird noch eine Matrix

$$\widetilde{\boldsymbol{W}} = \begin{pmatrix} \sum_{l=1}^{L} w_{11l} & \dots & \sum_{l=1}^{L} w_{1ql} \\ \vdots & & \vdots \\ \sum_{l=1}^{L} w_{n1l} & \dots & \sum_{l=1}^{L} w_{nql} \end{pmatrix},$$

welche analog zu $\boldsymbol{W}^{\star}$, jedoch ohne Berücksichtigung des gesamten Vektors $\boldsymbol{\gamma}$, konstruiert wird, benötigt, um Variablenselektion anhand der Größe

$$\|\widetilde{\boldsymbol{W}}_l^T(\boldsymbol{Y} - \boldsymbol{\pi}_{-l})\|_F, \quad i = 1, \dots, n,$$

durchführen zu können. Mit $\widetilde{\boldsymbol{W}}_l$ wird dabei die Matrix

$$\widetilde{\boldsymbol{W}}_l = \begin{pmatrix} w_{11l} & \dots & w_{1ql} \\ \vdots & & \vdots \\ w_{n1l} & \dots & w_{nql} \end{pmatrix}, \quad i = 1, \dots, n,$$

bezeichnet.

Die umformulierte Log-Likelihood (Gleichung (26)) aus Kapitel 4.4.1 kann um kategorienspezifische Einflussgrößen erweitert werden. Hierfür sind jedoch Vorbereitungen notwendig. Es wird eine zu $\boldsymbol{Y}$ entsprechende und an die globalen Koeffizienten $\boldsymbol{\gamma}$ angepasste Matrix

$$\boldsymbol{Y}^\star = \begin{pmatrix} \boldsymbol{y}_1^T \\ \vdots \\ \boldsymbol{y}_n^T \end{pmatrix}$$

konstruiert. Diese entspricht $\boldsymbol{Y}$ ohne Berücksichtigung der letzten, zur Referenzkategorie gehörenden Spalte. Aus dieser Matrix $\boldsymbol{Y}^\star$ wird eine blockdiagonale Matrix $\widetilde{\boldsymbol{Y}}$ konstruiert, deren Blöcke jeweils aus den Zeilen von $\boldsymbol{Y}^\star$ bestehen. Es gilt also

$$\widetilde{\boldsymbol{Y}} = \begin{pmatrix} \boldsymbol{y}_1^T & 0 & \dots & \\ 0 & \boldsymbol{y}_2^T & 0 & \dots \\ \vdots & 0 & \ddots & \\ & \vdots & & \boldsymbol{y}_n^T \end{pmatrix}$$

Definiert man nun noch die Matrix

$$\boldsymbol{Z} = \widetilde{\boldsymbol{Y}}\boldsymbol{W},$$

so kann die Log-Likelihood geschrieben werden als

$$l(\boldsymbol{\beta}) = \sum \operatorname{diag}(\boldsymbol{X}\widetilde{\boldsymbol{\beta}}^T\boldsymbol{Y}^T) + \sum \boldsymbol{Z}\boldsymbol{\gamma} - \sum_{i=1}^{n} \log(1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T\boldsymbol{\beta}_{s\cdot} + w_{is}^\star)),$$

wobei mit $w_{is}^\star$ der Eintrag in der i-ten Zeile und s-ten Spalte der Matrix $\boldsymbol{W}^\star$ bezeichnet wird. Die Zielfunktion in Abhängigkeit von $\boldsymbol{\beta}_{\_j}$, sprich $\widetilde{S}(\boldsymbol{\beta}_{\cdot j})$ (vgl. Gleichung 27), kann ohne große Umstellung auch bei Berücksichtigung von kategorienspezifischen Einflussgrößen bei festgehaltenem $\boldsymbol{\gamma}$ und $\boldsymbol{\beta}_{\cdot -j}$ formuliert werden. Analog hierzu wird die penalisierte Log-Likelihood in Abhängigkeit von γ_l, also $\widetilde{S}(\gamma_l)$ $(l = 1, \dots, L)$, geschrieben als

$$\begin{aligned} \widetilde{S}(\gamma_l) = &\sum \operatorname{diag}(\boldsymbol{X}\tilde{\boldsymbol{\beta}}^T\boldsymbol{Y}^T) + \sum \boldsymbol{Z}_{-l}\boldsymbol{\gamma}_{-l} + \sum \boldsymbol{z}_l\gamma_l \\ &- \sum_{i=1}^{n} \log(1 + \sum_{s=1}^{q} \exp(\boldsymbol{x}_i^T\boldsymbol{\beta}_s + w_{is,-l}^\star + w_{isl}^\star)) \\ &- s(q)\lambda \sum_{j=1}^{p} (\beta_{1j}^2 + \ldots + \beta_{qj}^2)^{1/2} - \lambda \sum_{l^\star \neq l} |\gamma_l^\star| - \lambda|\gamma_l|, \quad l = 1, \dots, L, \end{aligned}$$

wobei mit $w_{is,-l}^\star$ der Eintrag in der i-ten Zeile und s-ten Spalte der Matrix $\boldsymbol{W}_{-l}^\star$ und mit $w_{isl}^\star$ der Eintrag in der i-ten Zeile und s-ten Spalte der Matrix $\boldsymbol{W}_l^\star$, deren

Konstruktion analog zu derjenigen von $\boldsymbol{W}^{\star}_{-l}$ verläuft, bezeichnet wird. Außerdem wird mit $\boldsymbol{Z}_{-l}$ die Matrix $\boldsymbol{Z}$ nach Entfernen der l-ten Spalte und mit $\boldsymbol{z}_l$ die l-te Spalte von $\boldsymbol{Z}$ bezeichnet.

Sind in einem Datensatz also globale (metrische und/oder kategoriale) und kategorienspezifische Einflussgrößen vorhanden, kann folgender Algorithmus angewendet werden:

1. Bestimme einen Startvektor für $\boldsymbol{\beta}$
2. Berechne den Interceptvektor $\boldsymbol{\beta}_{.0}$ als $\arg\min_{\boldsymbol{\beta}_{.0}}\{\widetilde{S}_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\}$.
3. Für einen Durchlauf durch $g = 1,\ldots,G$ bzw. $l = 1,\ldots,L$ gehe folgendermaßen vor:
   (a) Für globale Einflussgrößen:
   - Falls $\|\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_F \leq s(q)\lambda$, setze $\boldsymbol{\beta}_{.g} = \boldsymbol{0}$.
   - Bestimme ansonsten $\boldsymbol{\beta}_{.g}$ über $\arg\min_{\boldsymbol{\beta}_{.g}}\{\widetilde{S}_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\}$.

   (b) Für kategorienspezifische Einflussgrößen:
   - Falls $\|\boldsymbol{W}_l^{\star}(\boldsymbol{Y} - \boldsymbol{\pi}_{-l})\|_F \leq \lambda$, setze $\boldsymbol{\gamma}_l = \boldsymbol{0}$.
   - Bestimme ansonsten $\boldsymbol{\gamma}_l$ über $\arg\min_{\boldsymbol{\gamma}_l}\{\widetilde{S}_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\})$
4. Wiederhole die Schritte 2 und 3 bis zur Konvergenz.

4.5 Alternative Variablenselektion

Wie bereits erläutert, findet im dritten Schritt des gezeigten Algorithmus, sowohl für den binären, als auch für den multinomialen Fall, die Variablenselektion statt. Dies geschieht für den j-ten Prädiktor ($j = 1,\ldots,p$) anhand der Überprüfung der Bedingung

$$\|\boldsymbol{X}_j^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-j}})\|_2 \leq \lambda$$

im Fall metrischer Einflussgrößen (vgl. Abschnitt 4.4.1) bzw. in der allgemeineren Form für den g-ten Prädiktor ($g = 1,\ldots,G$) über

$$\|\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_F \leq \lambda$$

bei einer kategorialen oder metrischen Einflussgröße g ($g = 1,\ldots,G$). Hierfür lässt sich jedoch auch in Anlehnung an Zahid und Tutz (2010, S. 8) als Alternative

$$\frac{\frac{1}{q\cdot \mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|}{\sum_{g=1}^{G}\frac{1}{q\cdot \mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|} = \frac{\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|}{\sum_{g=1}^{G}\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|} < \frac{1}{p}, \quad g = 1,\ldots,G, \tag{32}$$

verwenden. Sind im betrachteten Datensatz auch kategorienspezifische Einflussgrößen vorhanden, berechnet man für die l-te Variable ($l = 1, \ldots, L$)

$$\frac{\frac{1}{q}|\gamma_l|}{\frac{1}{q}\|\boldsymbol{\gamma}\|_1} = \frac{|\gamma_l|}{\|\boldsymbol{\gamma}\|_1} < \frac{1}{L}, \quad l = 1, \ldots, L. \tag{33}$$

Ist für eine Kovariable die entsprechende Bedingung erfüllt, so wird der dazugehörige/werden die dazugehörigen Koeffizient(en) auf Null gesetzt und der Prädiktor damit aus dem Modell genommen. Zwar taucht das λ in diesen Ungleichungen nicht mehr auf, jedoch hängen die β- und γ-Werte vom Shrinkage-Parameter weiterhin ab, wodurch die Variablenselektion wiederum beeinflusst wird. Diese Variablenselektion kommt derjenigen einer „Subset Selection“ (vgl. Hastie et al., 2009, S. 57 ff.) sehr nahe.

Bei Verwendung der hier vorgestellten Variablenselektion können sich zwei Problematiken ergeben:

1. Nenner und Zähler dieses Kriteriums können gleich Null sein. Dies geschieht bei sehr hohen λ-Werten, da hierbei die β-Werte stark geshrinkt werden. Dies wiederum bedeutet jedoch, dass im Endmodell alle β-Werte auf Null gesetzt sein sollten. Damit der Algorithmus auch in einem solchen Fall funktioniert, wird der Nenner

   $$\sum_{g=1}^{G} \frac{1}{\mathrm{df}_g} \sum_{r=1}^{q} \sum_{m=1}^{\mathrm{df}_g} |\beta_{rm}|$$

   bzw.

   $$\|\boldsymbol{\gamma}\|_1$$

   des Kriteriumswert, falls er gleich Null ist, auf ∞ gesetzt. Damit ist der gesamte Kriteriumswert gleich Null und auf jeden Fall kleiner als $\frac{1}{p}$.

2. Geschätzte Koeffizienten können gegen $\pm\infty$ gehen, sodass der entsprechende Zähler und auch der Nenner des/der entsprechenden Kriteriumsterme(s) gegen ∞ gehen. In einem solchen Fall wird der Zähler aus rein praktischen Gründen auf Null gesetzt, damit nicht ∞/∞ bzw. `Inf/Inf` (in `R`) berechnet werden muss, was in der Software `R`, anhand welcher der in Kapitel 4.6 vorgestellte Algorithmus umgesetzt wurde, zu einer Fehlermeldung und einem Abbruch der Berechnungen, die über die Funkion `plregr` veranlasst werden, führt. Mit dieser Maßnahme wird also sichergestellt, dass der Algorithmus trotz Konvergenzprobleme am Ende konkrete Schätzwerte liefert, welche schlechtestenfalls alle gleich Null sind. Diese beiden „Sonderbehandlungen“ werden auch in Kapitel 4.6 eingesetzt, ohne speziell darauf einzugehen.

Bei genauerer Betrachtung der Lösung der ersten Problematik wird jedoch deutlich, dass in dieser Form die Wahl eines Nullvektor für $\boldsymbol{\beta}$ bzw. $\boldsymbol{\gamma}$ als Startvektor

problematisch sein kann. Um dieses Problem wiederum zu umgehen, könnte bspw. diese alternative Variablenselektion in der ersten Iteration des Algorihtmus weggelassen und erst ab der zweiten Iteration eingesetzt werden.

Alternativ kann diese Selektion auch erst am Ende jeder Iteration geschehen. Dabei wird zunächst jeder Parameter anhand des in den Kapiteln 4.4 und 4.7 vorgestellten Algorithmus angepasst und anschließend entschieden, welche Variable im Modell enthalten bleibt. Genauso könnte man die Selektion auch erst nach allen zur Anpassung der Koeffizienten benötigten Iterationen durchführen.

4.6 Kombinierte Variablenselektion

Eine Kombination der in Kapitel 4.5 vorgestellten Methodik zur Variablenselektion mit der ursprünglichen Vorgehensweise (Kapitel 4.4) wird in der Funktion `plregr` durchgeführt und auch in den später gezeigten Simulationen und den Berechnungen bzgl. des realen Datensatzes angewendet. Es werden in einem Schritt die Bedingungen

$$\|\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_F \leq \lambda$$

und

$$\frac{\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|}{\sum_{g}^{G}\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|} < \frac{1}{p}, \quad g = 1, \ldots, G,$$

bzw.

$$\|\boldsymbol{W}_l^{\star}(\boldsymbol{Y} - \boldsymbol{\pi}_{-l})\|_F \leq \lambda$$

und

$$\frac{|\gamma_l|}{\|\boldsymbol{\gamma}\|_1} < \frac{1}{L}, \quad l = 1, \ldots, L,$$

parallel verwendet. Insgesamt ergibt sich damit folgender Algorithmus:

1. Bestimme einen Startvektor für $\boldsymbol{\beta}$
2. Berechne den Interceptvektor $\boldsymbol{\beta}_{.0}$ als $\arg\min_{\boldsymbol{\beta}_{.0}}\{\widetilde{S}_\lambda(\boldsymbol{\beta})\}$.
3. Für einen Durchlauf durch $j = 1, \ldots, p$ bzw. $l = 1, \ldots, L$ gehe folgendermaßen vor:
   (a) Für globale Einflussgrößen:
   - Falls die Bedingung

     $$\|\boldsymbol{X}_g^T(\boldsymbol{Y} - \boldsymbol{\pi}_{\boldsymbol{\beta}_{-g}})\|_F \leq s(q)s(\mathrm{df}_g)\lambda$$

     und/oder

     $$\frac{\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|}{\sum_{g=1}^{G}\frac{1}{\mathrm{df}_g}\sum_{r=1}^{q}\sum_{m=1}^{\mathrm{df}_g}|\beta_{rm}|} < \frac{1}{p}, \quad g = 1, \ldots, G,$$

     zutrifft, setze $\boldsymbol{\beta}_{.g} = \mathbf{0}$.

- Bestimme ansonsten $\boldsymbol{\beta}_{.g}$ über $\arg\min_{\boldsymbol{\beta}_{.g}}\{\widetilde{S}_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\}$.

(b) Für kategoriespezifische Einflussgrößen:

- Falls die Bedingung

$$\|\boldsymbol{W}_l^\star(\boldsymbol{Y}-\boldsymbol{\pi}_{-l})\|_F \leq \lambda$$

und/oder

$$\frac{|\gamma_l|}{\|\boldsymbol{\gamma}\|_1} < \frac{1}{L}, \quad l=1,\ldots,L,$$

zutrifft, setze $\boldsymbol{\gamma}_l = \mathbf{0}$.

- Bestimme ansonsten $\boldsymbol{\gamma}_l$ über $\arg\min_{\boldsymbol{\gamma}_l}\{\widetilde{S}_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\}$

4. Wiederhole die Schritte 2 und 3 bis zur Konvergenz.

In dieser Form wird der Algorithmus für die folgenden Analysen verwendet.

Bei Verwendung dieser Kombination mit der „alternativen“ Variablenselektion scheint man mit dem Algorithmus bei Testdurchläufen auf weniger Konvergenzprobleme zu stoßen. Dies wäre also eine Motivation dafür, diese Form des Algorithmus zu verwenden. Als Vergleich ist auf der beigefügten CD in dem Dokument „Algorithmus_ohne_Alt.r“ eine Version der Funktion `plregr`, in welcher der Algorithmus ohne die kombinierte Variablenselektion, also in der in den Kapiteln 4.4.1–4.4.3 vorgestellten Form, implementiert ist.

Zur Veranschaulichung soll, wie bereits in Kapitel 4.4.1 angekündigt, ein Beispiel für Pfadverläufe zu einem Datensatz gezeigt werden. Hierfür wird ein Datensatz mit 150 Beobachtungen, fünf Kovariablen, drei Reponsekategorien und multivariat standardnormalverteilten Kovariablenausprägungen verwendet. Das wahre $\boldsymbol{\beta}$ in der in Kapitel 4.4.1 eingeführten Matrixschreibweise lautet

$$\boldsymbol{\beta}^\star = \begin{pmatrix} 0 & -1 & 0{,}8 & 0 & 0 & 0 \\ 0 & -0{,}8 & 1 & 0 & 0 & 0 \end{pmatrix},$$

jedoch mit der ersten, statt der K-ten Responsekategorie als Referenzkategorie. Es wurde also ein Design gewählt, in welchem lediglich die ersten beiden Kovariablen einen von Null verschiedenen Effekt auf den Response aufweisen. Somit sollten die letzten drei Einflussgrößen vom Algorithmus nicht für das Modell berücksichtigt und somit die entsprechenden β-Werte auf Null geschätzt werden. Die maximale Anzahl an Optimierungsschritten innerhalb der Funktion `optim` wurde auf 500 und $\epsilon = 10^{-5}$ gesetzt (vgl. Simulation 1 für ein sehr ähnliches Design) Im Vergleich zu Simulation 1 aus Kapitel 5.1 wurden lediglich die letzten vier Spalten von $\boldsymbol{\beta}^\star$, welche ausschließlich Nullen enthalten, weggelassen. Ansonsten waren das Design und auch alle Parametereinstellungen (vgl. Kapitel 5.1) gleich. Zur Erstellung der Pfade wurde eine λ-Sequenz von 50 Werten, wie in Kapitel 5 dargestellt

(vgl. auch Abbildung 3), verwendet. Die Schätzung der Koeffizienten geschah, wie auch bei den Simulationen und bei der Anwendung auf die realen Daten über die Funktion `plregr`, in der Version, welche im Dokument „Algorithmus.r“ auf der dieser Arbeit beigefügten CD zu finden ist. Es sei an dieser Stelle angemerkt, dass in der Datei „Algorithmus.r“ noch nach der Berechnung der Schätzungen zu den Simulationen Veränderungen an der Kommentierung durchgeführt wurden, was jedoch auf die Ergebnisse keinerlei Einfluss haben sollte.

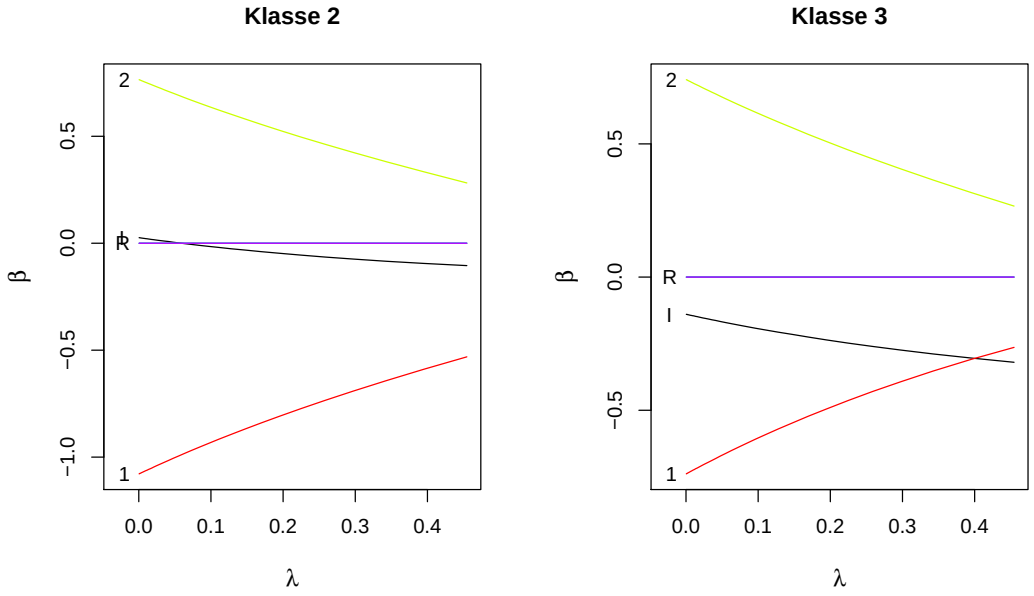

Abbildung 1: β-Pfade entlang einer Sequenz von λ-Werten aufgeteilt nach Responsekategorien; Die Beschriftung der Pfade beschreibt, zu welcher Einflussgröße diese jeweils gehören; „I“ steht für den Intercept und „R“ für die Variablen 3, 4 und 5.

In Abbildung 1 sind die Verläufe der Koeffizienten entlang dem λ-Pfad aufgeteilt nach den Responsekategorien zu sehen. Zu beobachten hierbei ist, dass die Pfade zu Variable 1 und 2 tendenziell für kleinere λ-Werte im Absolutwert größer werden. Die Variablen 3 – 5 verlassen während des gesamten Pfades nicht den Wert Null und würden somit unabhängig von der Wahl von λ aus dem Modell genommen werden.

In Abbildung 2, in welcher die Pfade nach Variablen aufgeteilt sind, wird nun deutlich, dass die Variablen nur für alle (in diesem Fall für beide) Responsekategorien beim gleichen λ-Wert in das Modell aufgenommen werden und somit nur gemeinsam selektiert werden können.

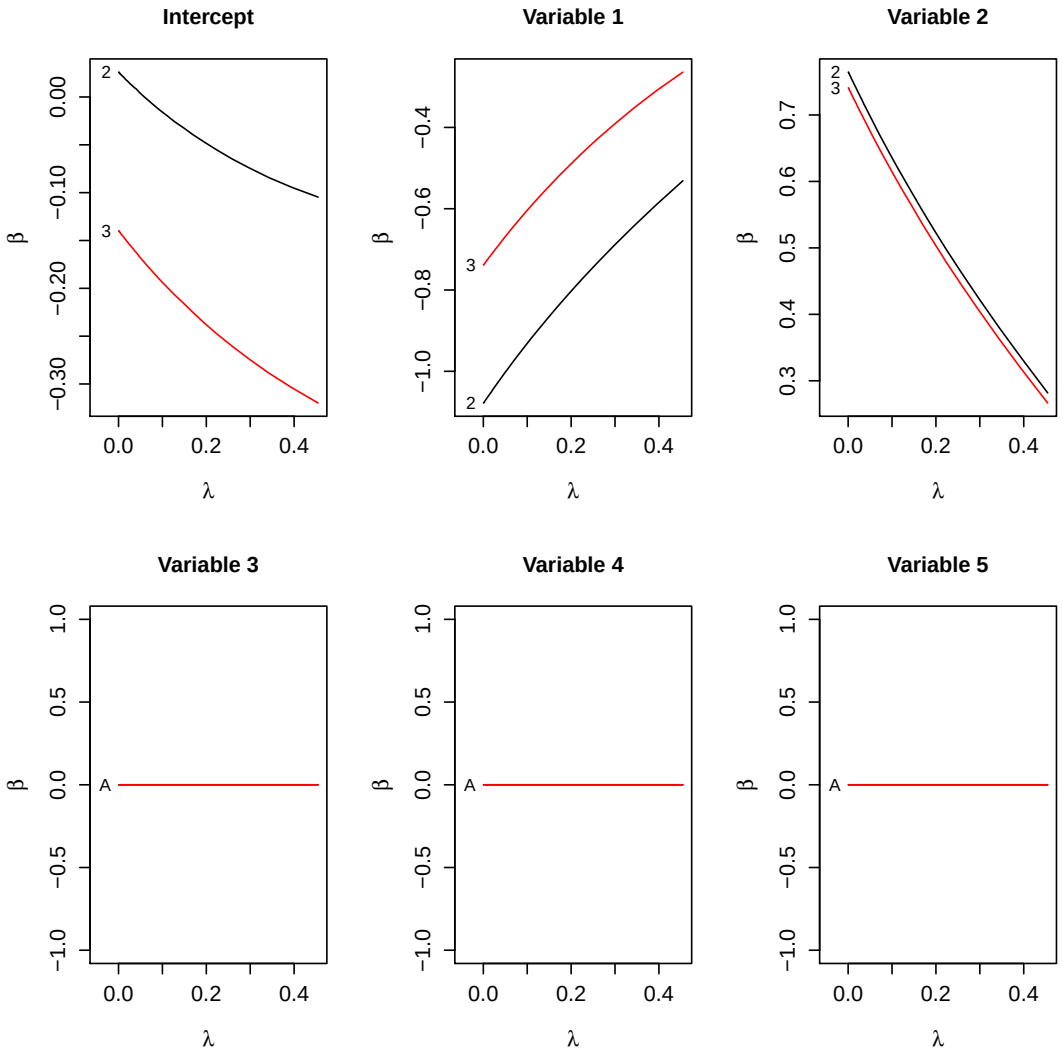

Abbildung 2: β-Pfade entlang einer Sequenz von λ-Werten aufgeteilt nach Kovariablen; Die Beschriftung der Pfade beschreibt zu welcher Responsekategorie diese jeweils gehören; „A“ in den unteren drei Plots steht für symbolisch für beide Klassen (man denke sich hierfür als Beschriftung bspw. „2, 3“)

4.7 Umsetzung des Algorithmus

Der Algorithmus wurde anhand der Software `R` umgesetzt. Eine Version der Funktionen hierzu ist in Anhang A zu finden. Für die Berechnung der Startwerte in Schritt 1 wird dabei, soweit nicht vom Anwender welche übergeben werden, entweder das Paket „`mlogit`“ (Croissant, 2010) oder das Paket „`nnet`“ (Venables und Ripley, 2002) verwendet. Für weitere Pakete, die beim Laden von `mlogit` ebenfalls geladen werden, sei auf Anhang A verwiesen. Zu beachten ist hierbei, dass

lediglich anhand der Funktion „`mlogit`“ aus dem gleichnamigen Paket Startwerte für kategorienspezifischen Einflussgrößen innerhalb der Funktion `plregr` berechnet werden können.

Im Algorithmus wird in jeder Iteration überprüft, ob die Log-Likelihood $\pm\infty$ oder den Wert Null annimmt. Dies führt ggf. dazu, dass für den gesamten Vektor $\hat{\boldsymbol{\beta}}$ bzw. die gesamte Matrix $\hat{\boldsymbol{\beta}}^{\star}$ Werte aus einer univariaten $N(0,1)$-Verteilung gezogen werden und von der Funktion `plregr` eine Warnmeldung ausgegeben wird. Mit $\hat{\boldsymbol{\beta}}$ und $\hat{\boldsymbol{\beta}}^{\star}$ werden die Schätzungen für $\boldsymbol{\beta}$ bzw. $\boldsymbol{\beta}^{\star}$ bezeichnet. Divergieren die Schätzer, werden also zufällige Werte eingesetzt, was i.d.R. dazu führen sollte, dass alle Koeffizientenwerte auf Null gesetzt werden. Im Normalfall sollte dies aufgrund der Selektion über das alternative Kriterium aus Kapitel 4.5 geschehen.

Den obigen Ausführungen entsprechend werden die einzelnen Schritte durchlaufen. Als Referenzkategorie wird hierbei nicht die K-te, sondern die erste Kategorie verwendet. Die Bestimmung von $\arg\min_{\boldsymbol{\beta}_{.0}}\{S_{\lambda}(\boldsymbol{\beta})\}$ im zweiten Schritt bzw. von $\arg\min_{\boldsymbol{\beta}_{.g}}\{S_{\lambda}(\boldsymbol{\beta})\}$ im dritten Schritt des Algorithmus wird über die Funktion „`optim`“ und der Einstellung „`method = "Nelder-Mead"`“, wodurch ein Nelder-Mead-Verfahren verwendet wird (s. die `R`-Hilfe zu `optim`), verwirklicht. Man gehe von M bis zur Konvergenz benötigten Iterationen aus. Hierbei wird pro Iteration nur eine begrenzte Anzahl an Optimierungsschritten durchgeführt. Es wird sich der Lösung ggf. also jeweils nur angenähert, da sich durch eine Hinzunahme weiterer Prädiktoren in der nächsten Iteration die Schätzsitutation möglicherweise ändert, wodurch die vorherige Annährung „veraltet“ sein kann. Standardmäßig werden in der Funktion `plregr` pro Iteration m ($m = 1, \ldots, M$) maximal zehn solcher Annäherungsschritte bei der jeweiligen Optimierung vollzogen. Über den Parameter „`maxiter`“ in der Funktion `plregr`, welcher wiederum den Parameter `maxit` in der Funktion `optim` ansteuert, lässt sich für die maximal durchzuführende Anzahl an Schritten eine entsprechende Einstellung tätigen. Durch eine Erhöhung des entsprechenden Wertes kann sich die Rechenzeit erhöhen. In den Simulationen und der Anwendung auf reale Daten in Kapitel 5 wird für `maxiter` in der Funktion `plregr` der Wert 500 verwendet, was den Standardeinstellungen bei Verwendung von `Nelder-Mead` in `optim` entspricht.

Bei einer Berechnung von Koeffizientenwerten entlang einer λ-Sequenz, werden beim ersten λ-Wert, $\lambda_{\min}$, je nach Parameterwahl in der Funktion `plregr`, Schätzungen aus der Funktion „`multinom`“ (Paket `nnet`) bzw. `mlogit` (Paket `mlogit`) als Startwerte für $\boldsymbol{\beta}$ und ggf. für $\boldsymbol{\gamma}$ in der ersten Iteration ($m = 1$) bei der Optimierung verwendet. Bei allen darauffolgenden λ-Werten werden die Schätzungen, die beim letzten λ-Wert als Endlösung (also nach Iteration $m = M$) berechnet wurden, als Startwerte benutzt. Für die späteren Iterationen ($m = 2, \ldots, M$) bzgl. eines λ-Wertes werden die Werte aus der vorhergehenden Iteration als Startwerte verwendet. M ist dabei i.A. für jeden λ-Wert und jeden Datensatz verschieden. Jedoch ist die maximale Anzahl an Iterationen standardmäßig auf 101 beschränkt. Wird die 101-te Iteration durchgeführt, so wird von der Funktion

`plregr` eine Warnmeldung ausgegeben, dass nach 100 Iterationen keine Konvergenz erzielt wurde. Über den Parameter „`maxiter.a`“ in der Funktion `plregr` lässt sich die maximale Anzahl dieser Iterationen jedoch verstellen. Im Allgemeinen werden in der Funktion `plregr` maximal `maxiter.a` + 1 solcher Iterationen durchgeführt. Die oben erwähnte Warnmeldung erfolgt, falls der (`maxiter.a` + 1)-te Durchgang benötigt wurde. Auf eine Warnmeldung bzgl. Konvergenzprobleme stößt man ggf. auch bei Berechnungen mit Hilfe der Funktion „`glmnet`“ aus dem gleichnamigen Paket „`glmnet`“ (Friedman et al., 2010). Daran orientiert sich das hier gewählte Vorgehen. Beim Laden des Paketes `glmnet` werden außerdem die Pakete „`Matrix`“ (Bates und Maechler, 2010) und „`lattice`“ (Sarkar, 2010) geladen.

Entsprechend wird für die Berechnung von $\arg\min_{\gamma_l}\{S_\lambda(\boldsymbol{\beta},\boldsymbol{\gamma})\}$ die Funktion „`optimize`“ verwendet. Konkrete Startwerte sind hierbei keine einsetzbar. Es ist jedoch jeweils ein Intervall nötig, innerhalb welchem nach dem Optimum gesucht wird. Dieses wird in der m-ten Iteration über

$$\left(\gamma_l^{(m-1)} - 2\cdot\sqrt{\hat{V}(\hat{\gamma}_l)},\quad \gamma_l^{(m-1)} + 2\cdot\sqrt{\hat{V}(\hat{\gamma}_l)}\right),\quad m = 2,\ldots,M,\quad l = 1,\ldots,L,$$

wobei mit $\gamma_l^{(m-1)}$ der Schätzwert aus der Iteration $m-1$ bezeichnet wird, bestimmt. Für $\hat{V}(\hat{\gamma}_l)$ wird das $(p+l)$-te Diagonalelement (pro Referenzkategorie werden p Koeffizientenvektoren angepasst) von

$$\hat{V}(\hat{\boldsymbol{\theta}}) = \left(\frac{\partial^2 l(\hat{\boldsymbol{\theta}})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\right)^{-1},$$

mit $\boldsymbol{\theta} = (\boldsymbol{\beta}^T,\boldsymbol{\gamma}^T)^T$ und $\hat{\boldsymbol{\theta}}$ als das geschätzte $\boldsymbol{\theta}$, eingesetzt (vgl. hierzu Hastie et al., 2009, S. 265 ff.). In der Funktion `plregr` wird dies, falls keine entsprechenden Werte übergeben werden, wovon hier ausgegangen wird, über den Rückgabewert „`hessian`“ der Funktion `mlogit`, anhand welcher ML-Schätzungen durchgeführt werden können, aus dem gleichnamigen Paket umgesetzt. Die Werte von $\hat{V}(\hat{\boldsymbol{\theta}})$ werden auch pro Schätzung nur einmal berechnet bzw. aus dem entsprechenden Argument, falls es vom Benutzer übergeben wird, in `plregr` übernommen und nicht in jeder Iteration neu bestimmt. Gleichermaßen verhält es sich bei einer Verwendung einer ganzen λ-Sequenz. Auch hier wird $\hat{V}(\hat{\boldsymbol{\theta}})$ insgesamt lediglich einmal (beim kleinsten λ-Wert) berechnet bzw. übernommen. Für die Berechnung von $\hat{V}(\hat{\boldsymbol{\theta}})$ wird also stets und unabhängig davon, in welcher Iteration und bei welchem λ-Wert innerhalb einer entsprechenden Sequenz man sich gerade befindet, der ML-Schätzer für $\hat{\boldsymbol{\theta}}$ verwendet.

Als Abbruchkriterium und somit zur Bestimmung von M wird

$$\|((\boldsymbol{\beta}_{\text{alt}}^{\star T} - \boldsymbol{\beta}_{\text{neu}}^{\star T}), (\boldsymbol{\gamma}_{\text{alt}}^T - \boldsymbol{\gamma}_{\text{neu}}^T))^T\|_F < \epsilon,$$

wobei $\boldsymbol{\beta}_{\text{alt}}^\star$ bzw. $\boldsymbol{\gamma}_{\text{alt}}$ für die entsprechende Koeffizientenmatrix bzw. den entsprechenden Koeffizientenvektor aus der vorhergehenden Iteration und $\boldsymbol{\beta}_{\text{neu}}^\star$ bzw. $\boldsymbol{\gamma}_{\text{neu}}$

für die aktuelle Koeffizientenmatrix bzw. den aktuellen Koeffizientenvektor stehen, verwendet. In der Funktion `plregr` kann das ϵ über den Parameter „`epsilon`“ bestimmt werden. Die Standardeinstellung hierfür ist 10^{-5}.

5 Anwendung des Algorithmus

Im Folgenden werden aus Simulationsstudien und der Anwendung auf einen realen Datensatz Ergebnisse, die anhand des im Kapitel 4.6 vorgestellten Algorithmus unter Verwendung der Funktion `plregr`, in der Version, welche in dem Dokument „Algorithmus.r“ auf der dieser Arbeit beigefügten CD enthalten ist, erzielt wurden, dargestellt. Es sei nochmals darauf hingewiesen, dass in der entsprechenden Datei „Algorithmus.r“ nach der Berechnung der in Kapitel 5.1 gezeigten Schätzungen noch Veränderungen, welche jedoch keinerlei Auswirkungen auf die Ergebnisse haben sollten, durchgeführt wurden.

5.1 Simulationsstudie

Der Algorithmus ist prinzipiell sowohl für binären, als auch für mehrkategorialen Response anwendbar. Bei binärem Response können jedoch bei der Berechnung über die Funktion `plregr` keine kategorienspezifischen Einflussgrößen berücksichtigt werden, was bei der Verwendung einer Responsekategorie jedoch auch keine wirkliche Einschränkung ist. Im Folgenden werden ausschließlich Datensätze mit drei Responsekategorien gezeigt und die Schätzungen, die aus dem Algorithmus gewonnen wurden, mit den ML-Schätzern, welche über die Funktion `mlogit` aus dem gleichnamigen Paket erziehlt wurden, verglichen. Hierzu wurde in Anlehnung an Zahid und Tutz (2009, S. 10 ff.) pro Iteration s ($s = 1, \ldots, S$) der MSE sowohl für $\hat{\boldsymbol{\pi}}$ als auch für $\hat{\boldsymbol{\beta}}$ und $\hat{\boldsymbol{\gamma}}$ berechnet. Hierfür wurden folgende Definitionen verwendet:

$$\mathrm{MSE}_s(\hat{\boldsymbol{\pi}}) = \frac{1}{qn} \sum_{i=1}^{n} \sum_{r=1}^{q} (\hat{\pi}_{ir} - \pi_{ir})^2 \quad s = 1, \ldots, S \tag{34}$$

bzw.

$$\mathrm{MSE}_s(\hat{\boldsymbol{\beta}}) = \|\hat{\boldsymbol{\beta}}_{s.} - \boldsymbol{\beta}\|_2^2 \quad s = 1, \ldots, S, \tag{35}$$

und

$$\mathrm{MSE}_s(\hat{\boldsymbol{\gamma}}) = \|\hat{\boldsymbol{\gamma}}_{s.} - \boldsymbol{\gamma}\|_2^2 \quad s = 1, \ldots, S, \tag{36}$$

wobei mit $\hat{\boldsymbol{\pi}}$ das geschätzte $\boldsymbol{\pi}$ bezeichnet wird (vgl. für eine entsprechende Darstellung Zahid und Tutz, 2009, S. 12 ff.). Der Index s wird im Folgenden zur Vereinfachung der Schreibweise unterdrückt. Zusätzlich wurden für die Schätzungen aus dem Algorithmus zwei weitere Kennzahlen berechnet. „False negatives“, die Anzahl der Kovariablen, deren Koeffizienten vom Algorithmus auf Null gesetzt werden, während die entsprechenden wahren Variablen von Null verschieden sind und „false positives“, die Anzahl der Kovariablen, deren Koeffizienten auf einen von Null verschiedenen Wert geschätzt werden, während die entsprechenden wahren Variablen gleich Null sind. Der Intercept wird hierbei nicht berücksichtigt. Man beachte zudem, dass diese Analysekriterien nach Kovariablen und nicht nach Parametern berechnet werden.

Um das „optimale“ λ zu bestimmen, wurde in jedem Durchgang eine λ-Sequenz berechnet, von welchen derjenige ausgewählt wurde, welcher den besten Wert bzgl. des Brier-Scores aus einer fünffachen Kreuzvalidierung lieferte.

Als höchster λ-Wert, bezeichnet mit $\lambda_{\max}$, wurde das Maximum über die Werte, welche sich aus $\|\boldsymbol{X}_g^T\boldsymbol{Y}\|_F$ bzw. $\|\boldsymbol{x}_g^T\boldsymbol{Y}\|_2$ dividiert durch das Gewicht $s(q)\cdot s(\mathrm{df}_g) = q^{1/2}\cdot g^{1/2}$, $(g = 1,\ldots,G)$, wobei $\boldsymbol{X}$ zuvor blockweise orthonormalisiert wurde, und $\|\boldsymbol{W}_l^\star\boldsymbol{Y}\|_F$ $(l = 1,\ldots,L)$, mit der oben gezeigten Transformation und blockweise orthonormalisierter Matrix $\boldsymbol{W}$, ergaben, verwendet. In Anlehnung an das Paper von Friedman et al. (2010, S. 8) wurde damit eine Sequenz vom Wert $\lambda_{\min} = \epsilon_\lambda \cdot \lambda_{\max}$ der Länge 50, mit $\epsilon_\lambda = 0.001$, bis hin zum dafür berechneten $\lambda_{\max}$ auf der Log-Skala gebildet (vgl. Friedman et al., 2010, S. 8). Die Abstände zwischen den einzelnen λ-Werten waren somit nicht gleich. Für die Erstellung einer λ-Sequenz wurde hier zunächst eine Folge mit äquidistanten Werten von $\log(0.001\cdot\lambda_{\max})$ bis $\log(\lambda_{\max})$ gebildet. Verwendet man dann diese Werte und transformiert sie zu $\exp(\lambda_{\min}) = \lambda_1, \lambda_2, \ldots, \exp(\lambda_{\max}) = \lambda_{50}$, so erhält man eine Sequenz, in welcher die Abstände anfangs sehr gering sind und mit wachsendem λ-Wert größer werden. Nach dieser Transformation sind $\lambda_{\min}$ und $\lambda_{\max}$ wieder an den Rändern zu finden. Zur Veranschaulichung vgl. man Abbildung 3, in welcher für $\lambda_{\max}$ der Wert fünf gewählt wurde.

In einem im Rahmen dieser Arbeit durchgeführten Vergleich von λ-Sequenzen, welche über das hier dargestellte Verfahren bestimmt wurden, mit denjenigen, welche über die Funktion `glmnet` gewonnen wurden, zeigte sich, dass zwar nicht genau dieselben Sequenzen erzeugt wurden, jedoch die jeweiligen Strukturen sehr ähnlich waren.

Es sollen noch weitere Schreibweisen erwähnt werden. „`maxiter.opt`“ bezeichnet die in der `R`-Funktion `optim` festgelegte maximale Anzahl der Schritte (vgl. Parameter `maxiter` in Kapitel 4.7) und ϵ_{opt} der entsprechende ϵ-Wert, welche bei der Optimierung während der Auswahl des „optimalen“ λ-Wertes über eine Kreuzvalidierung Verwendung finden. Die Optimierung fand über die Funktion `plregr.cv`, in welcher entsprechende Parameter zu finden sind, statt. `maxiter` und ϵ selbst wird von nun an dafür verwendet, um entsprechende Parametereinstellungen (`maxiter` und `epsilon` in der `R`-Funktion `plregr`), die bei der Optimierung während der Bestimmung des entgültigen Modells angewandt werden, zu beschreiben. In allen Simulationen wurde `maxiter.opt` = `maxiter` = 500 und $\epsilon_{\text{opt}} = \epsilon = 10^-5$ (`1e-5` in `R`) gesetzt. Außerdem wurde ebenfalls in allen Simulationen die maximale Anzahl an Iterationen über alle Kovariablen hinweg auf 101 festgelegt (vgl. hierzu den Parameter `maxiter.a` in der Funktion `plregr`). Zu beachten ist, dass in der Funktion `plregr` maximal `maxiter.a` + 1 Iterationen durchgeführt werden. Es werden also zunächst `maxiter.a` Iterationen durchgeführt, um anschließend, falls hierbei keine Konvergenz erziehlt wurde, nochmals einen abschließenden Durchgang zu berechnen. In den hier gezeigten Simulationen wurde die Standardeinstellung von `maxiter` (= 100) also nicht verändert.

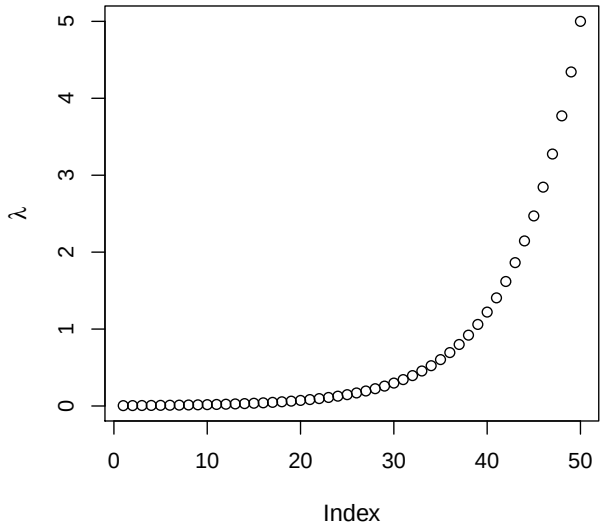

Abbildung 3: Beispiel für eine λ-Sequenz der Länge 50 zur Bestimmung des „optimalen“ Werts für λ mit $\lambda_{\max} = 5$

Pro Simulationsstudie wurden $S = 20$ Durchläufe mit der gleichen zugrunde liegenden Datenstruktur berechnet. Die Startwerte wurden hierbei, sowie bei den Berechnungen zum realen Datensatz in Kapitel 5.2, über das Paket `mlogit` bestimmt.

Die $\boldsymbol{\beta}$-Vektoren werden, um eine übersichtlichere Darstellung zu erhalten, über $\boldsymbol{\beta}^\star$ (vgl. Kapitel 4.4.1) dargestellt. $\boldsymbol{\beta}^\star$ besitzt die Dimension $q \times (p+1)$, wobei pro Spalte ein $\boldsymbol{\beta}_{.j}$ $(j = 1, \ldots, p)$ zu finden ist. In den Simulationen wurde außerdem der wahre Intercept immer auf Null gesetzt. Bei der Berechnung des jeweiligen $\text{MSE}(\hat{\boldsymbol{\beta}})$ wurde dieser ebenfalls berücksichtigt.

Zur Generierung der Daten fand die Funktion „`sim`“, welche auf der beigefügten CD zu dieser Arbeit in der Datei „Generierung.r“ zu finden ist, Verwendung. Hierbei wurde das Paket „`MASS`“ (Venables und Ripley, 2002) verwendet. Der Code, welcher zur Durchführung der Simulationen verwendet wurde, ist ebenfalls auf der beigefügten CD zu finden. Die entsprechende Datei ist mit „Simulationen.r“ bezeichnet und enthält, neben ergänzenden Kommentaren, auch Code, anhand welchem eine zu den hier gezeigten Auswertungen sehr ähnliche grafische Analyse der Ergebnisse durchgeführt werden kann.

Simulation 1: Die Generierung der Werte für die Prädiktoren fand über eine multivariate Normalverteilung ($N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, mit $\mu = \mathbf{0}$ und $\boldsymbol{\Sigma} = I_{10}$) statt. Die Anzahl der metrischen Einflussgrößen wurde für jede der $n = 150$ Beobachtung

auf $p = 10$ gesetzt. In Matrixdarstellung wurden für $\boldsymbol{\beta}$ folgende Werte verwendet:

$$\boldsymbol{\beta}^{\star}_{2\times 11} = \begin{pmatrix} 0 & -1{,}0 & 0{,}8 & 0 & \dots \\ 0 & -0{,}8 & 1{,}0 & 0 & \dots \end{pmatrix}$$

Somit hatten lediglich die ersten zwei Kovariablen Einfluss auf den Response und sollten daher vom Algorithmus auch in das Modell aufgenommen werden, während möglichst alle anderen Prädiktoren keine Berücksichtigung erhalten sollten.

Ergebnisse: Wie bereits oben erläutert werden die Ergebnisse aus den Schätzungen des Algorithmus und die ML-Schätzer anhand des MSE($\hat{\boldsymbol{\pi}}$) und des MSE($\hat{\boldsymbol{\beta}}$) analysiert. Abbildung 4 zum MSE($\hat{\boldsymbol{\beta}}$) zeigt recht deutliche Vorteile des Algorithmus gegenüber dem ML-Schätzer, was aufgrund des gewählten Designs und der Tatsache, dass ML-Schätzungen, zumindest theoretisch, nicht exakt Null sein können, nicht allzu überraschend sein dürfte. Somit zeigt sich auch bzgl. des MSE($\hat{\boldsymbol{\pi}}$) ein besseres Ergebnis auf Seiten des Algorithmus (vgl. Abbildung 5). Ein interessantes Bild liefern jedoch auch die Grafiken zu den false negatives und false positives (Abbildung 6). Hier zeigt sich, dass stets, mit einer Ausnahme, die Variablen mit wahrem Einfluss auf den Response erkannt und angepasst wurden, während zwar in manchen Durchläufen eine oder zwei Variable(n) fälschlicherweiße in das Modell zusätzlich aufgenommen wurden, dies jedoch nicht allzu häufig passierte.

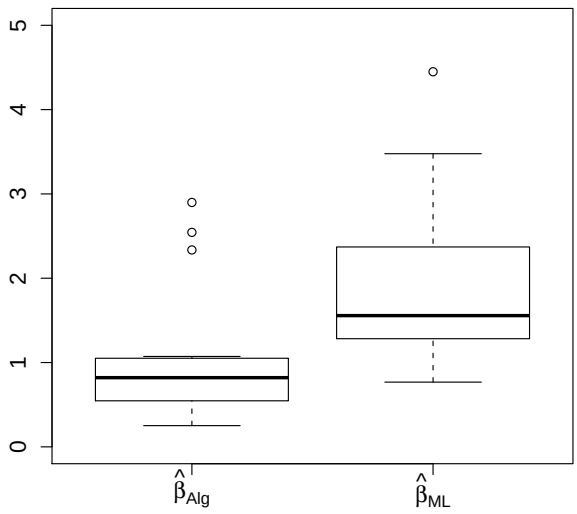

Abbildung 4: Vergleich der MSE($\hat{\boldsymbol{\beta}}$) aus Simulation 1; links: aus Algorithmus ($\hat{\beta}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\beta}_{\text{ML}}$)

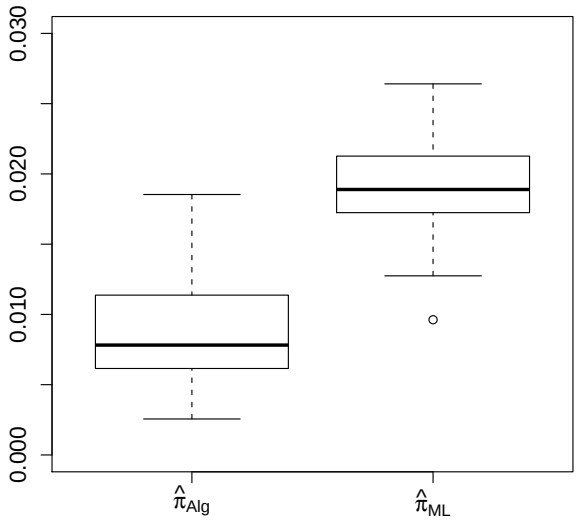

Abbildung 5: Vergleich der MSE($\hat{\boldsymbol{\pi}}$) aus Simulation 1; links: aus Algorithmus ($\hat{\pi}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\pi}_{\text{ML}}$)

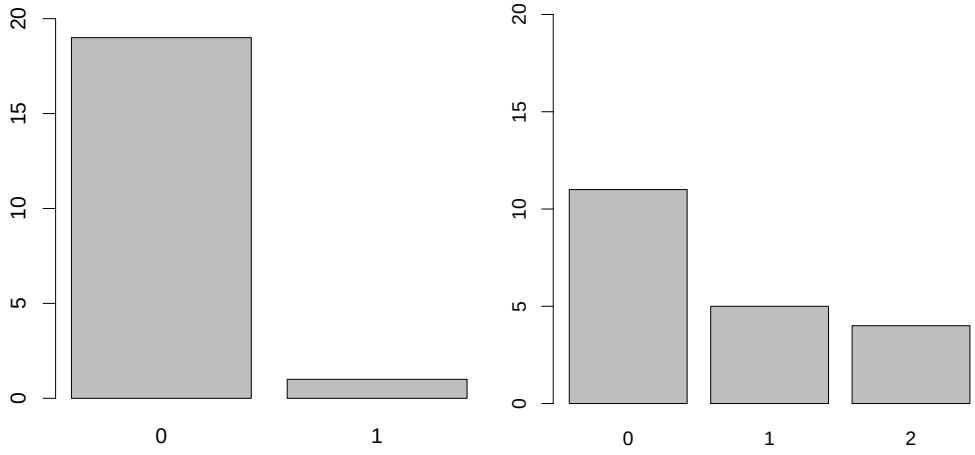

Abbildung 6: False negatives (links) und false positives (rechts) des Algorithmus aus Simulation 1

Simulation 2: Eine weitere Studie bestand darin, ebenfalls die Werte zu $p = 10$ metrischen Kovariablen über eine $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$-Verteilung, mit $\mu = \mathbf{0}$ zu genieren. Jedoch sollte für $\boldsymbol{\Sigma}$ die autokorrelierte Kovarianzmatrix

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & \rho & \rho^2 & \rho^3 & \dots & & & \\ \rho & 1 & \rho & \rho^2 & \rho^2 & \dots & & \\ \rho^2 & \rho & 1 & \rho & \rho^2 & \rho^3 & \dots & \\ \rho^3 & \rho^2 & \rho & 1 & \rho & \rho^2 & \rho^3 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix},$$

mit $\rho = 0{,}5$, Anwendung finden. Zudem galt

$$\boldsymbol{\beta}^{\star}_{2\times 11} = \begin{pmatrix} 0 & -1{,}2 & 0{,}7 & 0 & \dots \\ 0 & 0{,}8 & -1{,}1 & 0 & \dots \end{pmatrix}.$$

Somit hatten hier, im Gegensatz zur Simulation 1, die Koeffizienten zu einer Kovariable verschiedene Vorzeichen. Die Referenzkategorie lag somit „zwischen“ den übrigen Responsekatogorien. Außerdem wurde $n = 300$ gesetzt.

Ergebnisse: Es zeigt sich ein sehr ähnliches Bild, wie in Simulation 1. Der Algorithmus liefert, gemessen an den MSE-Werten (vgl. Abbildungen 7 und 8), tendenziell bessere Ergebnisse. Hierbei sei angemerkt, dass in Abbildung 7 ein Ausreißer beim Boxplot zum $\mathrm{MSE}(\hat{\beta}_{\mathrm{ML}})$ entfernt wurde. Dieser lag bei einem gerundeten Wert von 3,39. Zu beachten ist nun allerdings, dass, im Gegensatz zur Simulation 1, keine false negatives produziert wurden. Das Balkendiagramm zu den false positives verhält sich sehr ähnlich zu dem aus Simulation 1 (s. Abbildung 9).

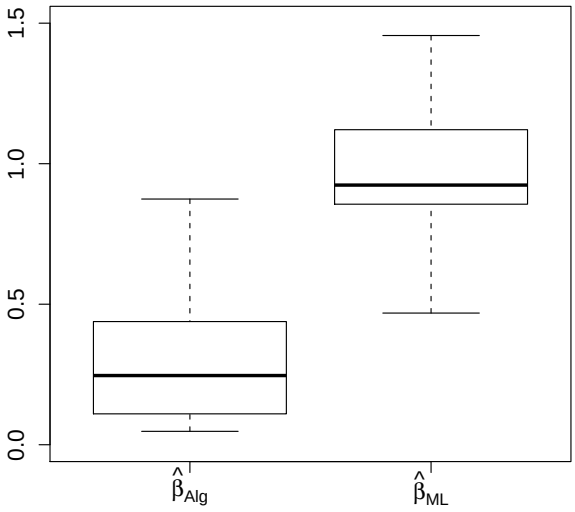

Abbildung 7: Vergleich der MSE($\hat{\boldsymbol{\beta}}$) aus Simulation 2; links: aus Algorithmus ($\hat{\beta}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\beta}_{\text{ML}}$); Beim rechten Boxplot zum MSE($\hat{\beta}_{\text{ML}}$) wurde ein Ausreißer mit dem gerundeten Wert 3,39 entfernt.

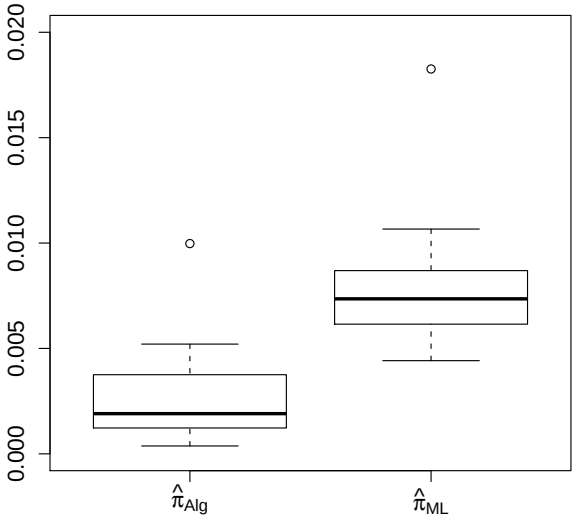

Abbildung 8: Vergleich der MSE($\hat{\boldsymbol{\pi}}$) aus Simulation 2; links: aus Algorithmus ($\hat{\pi}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\pi}_{\text{ML}}$)

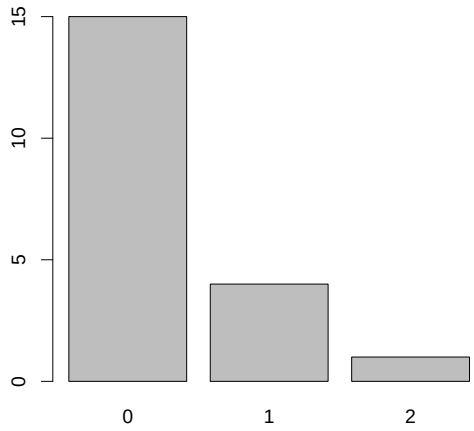

Abbildung 9: false positives des Algorithmus aus Simulation 2

Simulation 3: Da bisher lediglich Einflussgrößen von metrischem Skalenniveau Verwendung fanden, wurden im folgenden Design ausschließlich kategoriale herangezogen. Die Generierung kategorialer Einflussgrößen wurde, genauso wie auch in den weiteren Simulationen, in Anlehnung an Yuan und Lin (2006, S. 60) durchgeführt. Hierzu wurden für $n = 200$ Beobachtungen $G = 5$ Prädiktoren ebenfalls über eine multivariate Normalverteilung generiert, jedoch mit zusätzlicher Einteilung der Werteskala in Kategorien. Zunächst wurden also die Parameter $\boldsymbol{\mu}_{\text{kat}}$ und $\boldsymbol{\Sigma}_{\text{kat}}$ definiert, anhand welcher dann $n = 200$ $N(\boldsymbol{\mu}_{\text{kat}}, \boldsymbol{\Sigma}_{\text{kat}})$-verteilte Werte gezogen wurden. Diese wurden pro Variable anschließend gemäß der Anzahl der Kategorien der jeweiligen Variable in Gruppen eingeteilt, deren Grenzen sich aus entsprechenden Quantilen ergaben. Man betrachte als Beispiel eine Kovariable mit 4 Kategorien. Aus der normalverteilten Datenmatrix lassen sich die zum entsprechenden Prädiktor gehörenden Werte in einen Vektor schreiben. Dieser Vektor wird nun in Gruppen unterteilt. Für die Grenzen werden das 1/4-, 2/4- und 3/4-Quantil, im Allgemeinen das $1/\text{df}_g$-, $2/\text{df}_g$-, ... und $(\text{df}_g - 1)/\text{df}_g$- Quantil $(g = 1, \ldots, G)$ verwendet. Diese Gruppenzugehörigkeiten werden als Ausprägungen der Kovariablen verwendet. Bzgl. der Zuteilung von ursprünglich aus der Normalverteilung generierten Beobachtungen, welche bei der Unterteilung des Wertebereichs an den Rändern der jeweiligen Intervalle liegen, sei auf die Funktion „`sim`“ (s. die beigefügte CD), welche bei der Generierung von kategorialen Kovariablen auf die Funktionen „`quantile`“ und „`cut`“ (s. die entsprechenden `R`-Hilfen) zurückgreift, verwiesen.

Für diese Studie wurde $\boldsymbol{\mu}_{\text{kat}} = \mathbf{0}$ und $\boldsymbol{\Sigma}_{\text{kat}} = I_5$, analog zur ersten Simulation, wobei für jede der G Kovariablen drei Kategorien verwendet wurden, eingesetzt. Als Koeffizientenmatrix diente

$$\boldsymbol{\beta}^{\star}_{2\times 11} = \begin{pmatrix} 0 & -0{,}9 & 1{,}1 & 0 & \ldots \\ 0 & -0{,}8 & 1{,}0 & 0 & \ldots \end{pmatrix}.$$

Die zweite und dritte Spalte beschreiben die Koeffizienten zur ersten Kovariable, welche als einzige einen von Null verschiedenen Effekt aufwies.

Ergebnisse: Auch in diesem Design sind Vorteile auf Seiten des Algorithmus sowohl bzgl. des MSE($\boldsymbol{\beta}$) als auch bzgl. des MSE($\boldsymbol{\pi}$) (s. Abbildung 10 bzw. 11) zu erkennen. Während auch in dieser Simulation keine false negatives erzeugt wurden, wird aus Abbildung 12 ersichtlich, dass hier die bisher meisten false negatives zu beobachten waren.

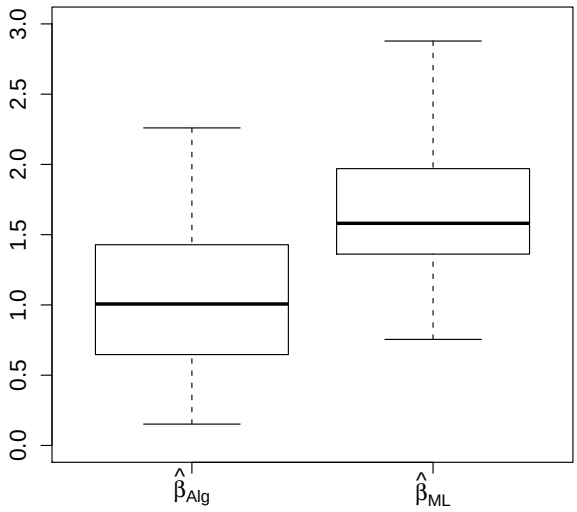

Abbildung 10: Vergleich der MSE($\hat{\boldsymbol{\beta}}$) aus Simulation 3; links: aus Algorithmus ($\hat{\beta}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\beta}_{\text{ML}}$)

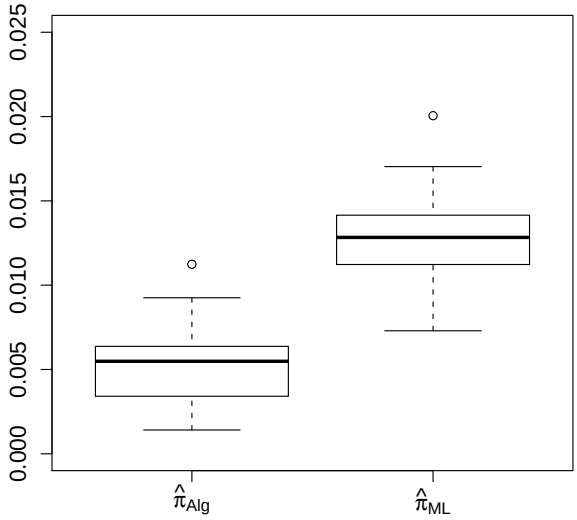

Abbildung 11: Vergleich der MSE($\hat{\boldsymbol{\pi}}$) aus Simulation 3; links: aus Algorithmus ($\hat{\pi}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\pi}_{\text{ML}}$)

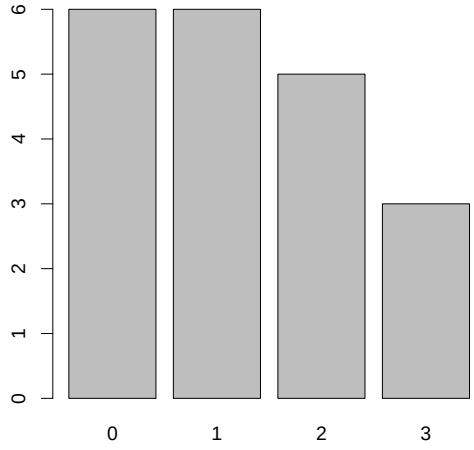

Abbildung 12: false positives des Algorithmus aus Simulation 3

Simulation 4: Anhand dieser Studie sollte nun eine Kombination von metrischen und kategorialen Prädiktoren eingesetzt werden. Hierzu wurden pro Durchlauf und Beobachtung ($n = 200$) 6 metrische und zwei kategoriale (mit jeweils drei Kategorien) Einflussgrößen generiert. Es wurde, analog zur Simulation 1 bzw. 3, $\boldsymbol{\mu} = \mathbf{0}$, $\boldsymbol{\mu}_{\text{kat}} = \mathbf{0}$, $\boldsymbol{\Sigma} = I_6$ und $\boldsymbol{\Sigma}_{\text{kat}} = I_2$ gewählt. Zudem wurde

$$\boldsymbol{\beta}^{\star}_{2\times 11} = \begin{pmatrix} 0 & -0{,}9 & 1{,}25 & 0 & \dots & -0{,}80 & 1{,}45 & 0 & 0 \\ 0 & -0{,}8 & 1{,}20 & 0 & \dots & -0{,}75 & 1{,}40 & 0 & 0 \end{pmatrix}$$

verwendet. Hierbei gehören die ersten sechs Spalten nach der zum Intercept gehörenden (also Spalte 2 – 7 von links) zu den metrischen Einflussgrößen. Die restlichen Spalten beschreiben den wahren Einfluss der kategorialen Kovariablen. Die den kategorialen Prädiktoren zugeordneten β-Werte bilden die Matrix

$$\begin{pmatrix} -0{,}80 & 1{,}45 & 0 & 0 \\ -0{,}75 & 1{,}40 & 0 & 0 \end{pmatrix}.$$

Nach den wahren β-Werten hatten also lediglich die ersten beiden metrischen Kovariablen und die erste kategoriale Kovariable einen von Null verschiedenen Effekt auf den Response.

Ergebnisse: In den Abbildungen 13 und 14 zeigt sich ein bereits bekanntes Bild. Hierbei wurden jedoch insgesamt drei Ausreißer entfernt. Einer davon beim Boxplot zum $\text{MSE}(\hat{\boldsymbol{\beta}}_{\text{Alg}})$ mit dem gerundeten Wert 5,61 und zwei beim Boxplot zum $\text{MSE}(\hat{\boldsymbol{\beta}}_{\text{ML}})$ mit den gerundeten Werten 4,27 und 4,26. Auch bzgl. der false positives und false negatives schneidet der Algorithmus sehr gut ab (vgl. Abbildung 15). Nur jeweils einmal wird ein β-Wert fälschlicherweiße auf Null gesetzt bzw. in das Modell aufgenommen. Dies geschah nicht in demselben Durchgang der Schätzungen.

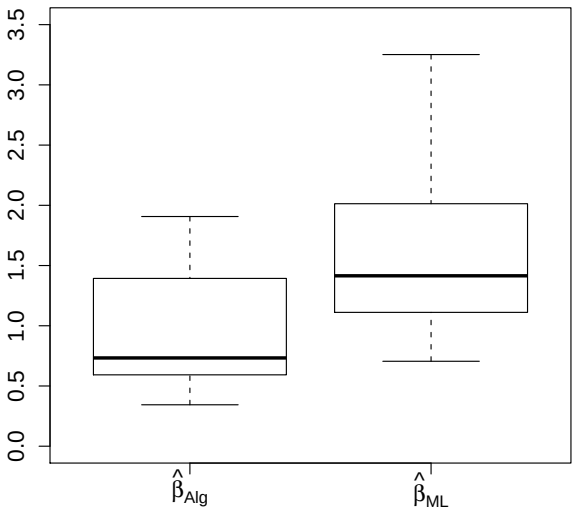

Abbildung 13: Vergleich der MSE($\hat{\boldsymbol{\beta}}$) aus Simulation 4; links: aus Algorithmus ($\hat{\beta}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\beta}_{\text{ML}}$); Insgesamt drei Ausreißer wurden hierbei entfernt: Einer beim Boxplot zum MSE($\hat{\boldsymbol{\beta}}_{\text{Alg}}$) mit dem gerundeten Wert 5,61 und zwei beim Boxplot zum MSE($\hat{\boldsymbol{\beta}}_{\text{ML}}$) mit den gerundeten Werten 4,27 und 4,26.

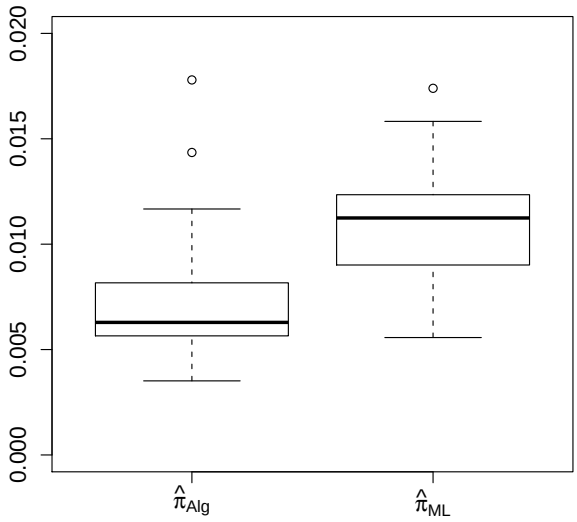

Abbildung 14: Vergleich der MSE($\hat{\boldsymbol{\pi}}$) aus Simulation 4; links: aus Algorithmus ($\hat{\pi}_{\mathrm{Alg}}$), rechts: aus ML-Schätzung ($\hat{\pi}_{\mathrm{ML}}$)

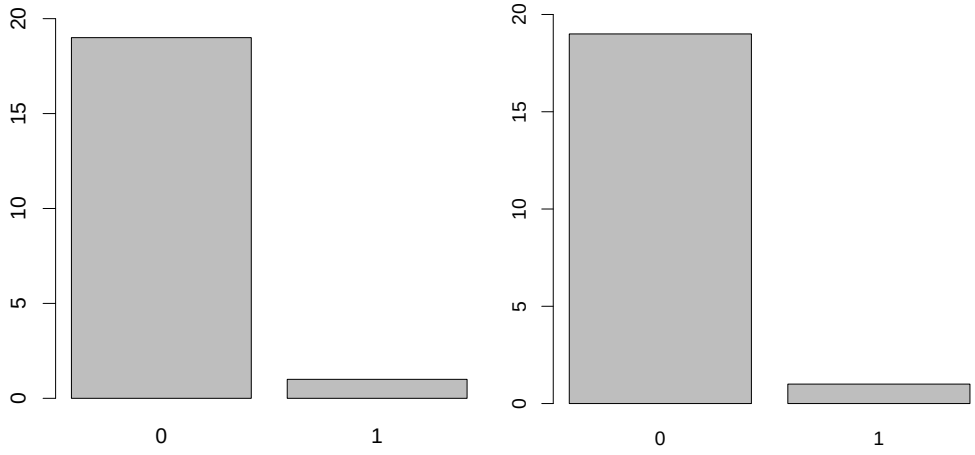

Abbildung 15: false negatives (links) und false positives (rechts) des Algorithmus aus Simulation 4

Simulation 5: Für die letzte hier gezeigte Simulation wurden nun alle behandelten Arten von Kovariablen miteinbezogen. Somit kamen im Vergleich zur letzten Simulation kategorienspezifische Einflussgrößen hinzu. Für die Simulation solcher Variablen werden zunächst $n \cdot K$ Werte aus einer $N(\boldsymbol{\mu}_{\text{ga}}, \boldsymbol{\Sigma}_{\text{ga}})$-Verteilung gezogen. Um sie direkt in den Algorithmus einsetzen zu können, werden in der Funktion `plregr` pro kategorienspezifischer Variable und Beobachtung die übrigen Ausprägungen von derjenigen der Referenzkategorie abgezogen.

In dieser Studie wurden fünf metrische, zwei kategoriale mit jeweils drei Kategorien und fünf kategorienspezifische Variablen verwendet. Für die Konstruktion der gesamten Datensätze wurde $\boldsymbol{\mu} = \mathbf{0}$, $\boldsymbol{\mu}_{\text{kat}} = \mathbf{0}$, sowie $\boldsymbol{\mu}_{\text{ga}} = \mathbf{0}$ und für $\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma}_{\text{kat}}$, sowie $\boldsymbol{\Sigma}_{\text{ga}}$, analog zu $\boldsymbol{\Sigma}$ aus Simulation 2, eine autokorrelierte Kovarianzmatrix, hier jedoch mit $\rho = 0{,}3$, gewählt. Die Koeffizienten waren wie in Simulation 5 über

$$\boldsymbol{\beta}^{\star}_{2\times 11} = \begin{pmatrix} 0 & -0{,}9 & 1{,}25 & 0 & \dots & -0{,}80 & 1{,}45 & 0 & 0 \\ 0 & -0{,}8 & 1{,}20 & 0 & \dots & -0{,}75 & 1{,}40 & 0 & 0 \end{pmatrix},$$

wobei wiederum in den ersten Spalten (von links) die Koeffizienten für die metrischen und anschließend diejenigen für die kategorialen Einflussgrößen beschrieben werden, bestimmt. Die Matrix aus den β-Werten bzgl. der kategorialen Kovariablen war somit

$$\begin{pmatrix} -0{,}80 & 1{,}45 & 0 & 0 \\ -0{,}75 & 1{,}40 & 0 & 0 \end{pmatrix}.$$

Pro Iteration wurden 450 Beobachtungen generiert. Als γ-Vektor wurde

$$\boldsymbol{\gamma} = \begin{pmatrix} 0{,}9 \\ 1{,}0 \\ 0{,}0 \\ 0{,}0 \\ 0{,}0 \end{pmatrix}$$

verwendet. Es waren also lediglich die ersten beiden wahren γ-Werte von Null verschieden.

Zudem sei hier angemerkt, dass die $S = 20$ Durchgänge in vier Blöcke zu je fünf Durchläufen aufgeteilt und getrennt voneinander berechnet wurden. Für Details sei auf die Datei „Simulationen.r“ auf der beigefügten CD verwiesen.

Ergebnisse: Die Boxplots zu den MSEs in den Abbildungen 16 und 17 bzgl. den β- und γ-Werten zeigen, dass auch hier der Algorithmus tendenziell etwas besser abschneidet. In Abbildung 18 ist zu sehen, dass die Wahrscheinlichkeiten scheinbar ebenfalls über den Algorithmus tendenziell besser schätzbar sind. Interessant ist noch, dass sowohl bzgl. $\boldsymbol{\beta}$ als auch bzgl. $\boldsymbol{\gamma}$ weder false negatives noch false positives erzeugt wurden.

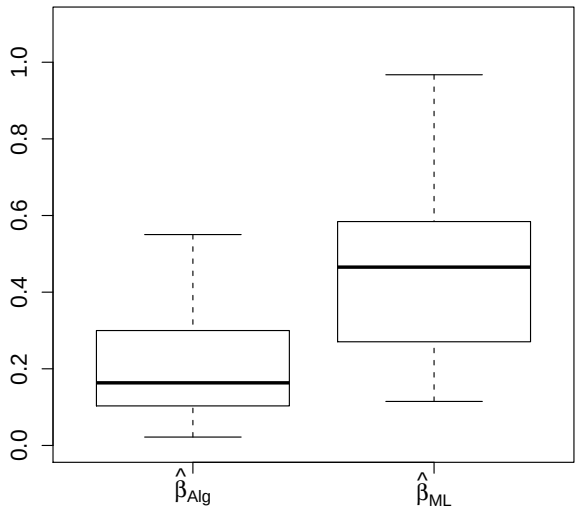

Abbildung 16: Vergleich der MSE($\hat{\boldsymbol{\beta}}$) aus Simulation 5; links: aus Algorithmus ($\hat{\beta}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\beta}_{\text{ML}}$)

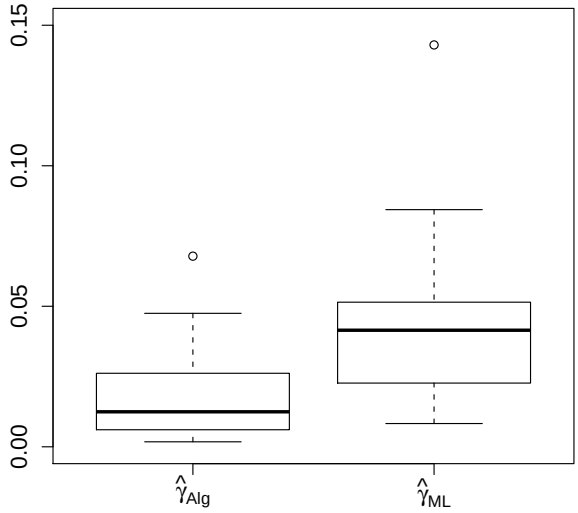

Abbildung 17: Vergleich der MSE($\hat{\boldsymbol{\gamma}}$) aus Simulation 5; links: aus Algorithmus ($\hat{\gamma}_{\text{Alg}}$), rechts: aus ML-Schätzung ($\hat{\gamma}_{\text{ML}}$)

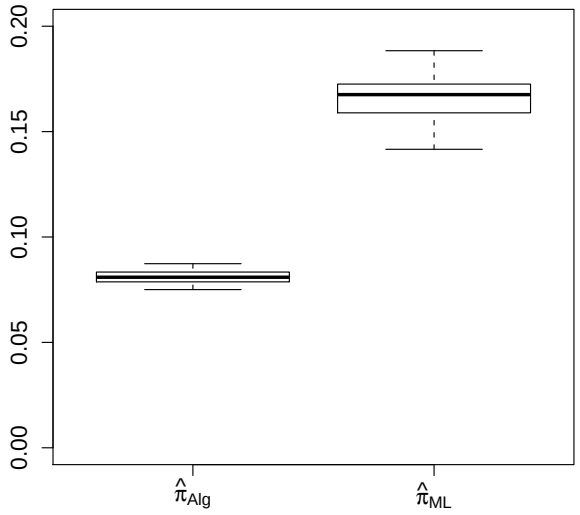

Abbildung 18: Vergleich der MSE($\hat{\boldsymbol{\pi}}$) aus Simulation 5; links: aus Algorithmus ($\hat{\pi}_{\mathrm{Alg}}$), rechts: aus ML-Schätzung ($\hat{\pi}_{\mathrm{ML}}$)

5.2 Reale Daten

In diesem Abschnitt wird der Algorithmus für das regularisierte multinomiale Modell auf einen Datensatz zur Analyse von politischen Wahlen mit globalen metrischen und kategorialen (wie z.B. Alter bzw. Geschlecht der Befragten), sowie kategorienspezifischen (metrischen) Einflussgrößen angewandt. Eine Beschreibung eines bzgl. der Struktur ähnlichen Datensatzes ist im Paper von Thurner und Eymann (2000, S. 64 f. bzw. S. 76 f.) zu finden. Neben der Wahlabsicht sind im hier behandelten Datensatz Informationen zur jeweiligen Person und mehrere „Scores“, welche zu verschiedenen Themengebieten die „Position“ der/des Befragten wiederspiegeln sollen, zu finden. Zu jedem dieser Bereiche wurden auch Scores zu den einzelnen Parteien gebildet, von welchen dann pro Person wiederum die Differenz gebildet wurde. Diese Differenzen wurden, neben den globalen Einflussgrößen, für die hier gezeigte Analyse verwendet. Responsevariable war die Angabe der Wahlabsicht der jeweiligen Person (Variable „Wa“; Im Dokument „VariablenDef BTW09small.docx“ (s. u.) ist lediglich eine Variable „WA“ zu finden; Es wurde angenommen, dass damit „Wa“ gemeint war). Die entsprechenden Parteien waren „CDU / CSU“, „SPD“, „FDP“, „Die Grünen“, und „Die Linke“. Die kategorienspezifischen Scores wurden bzgl. den Themen „sozioökonomische Dimension“, „Zuzug von Ausländern“, „Kernenergie“ und „Links-Rechts-Einstufung“ gebildet, wobei die Kovariable zur Position bzgl. des Scores zur Links-Rechts-Einstufung der CDU zumindest im hier behandelten Datensatz fehlt. Die Differenzen wurden pro Befragten zudem sowohl zur CSU also auch zur CDU berechnet. Bei der Analyse wurde dies dahingehend berücksichtigt, als dass lediglich die Differenzen zur CSU verwendet wurden, falls die Person aus Bayern stammte, und sonst die zur CDU. Die globalen Einflussgrößen waren „Gewerk“ (1: Gewerkschaftsmitgliedschaft, 0: sonst), „West“ (1: Westdeutschland, 0: Ostdeutschland), „Geschlecht“ (1: männlich, 0: weiblich), „Bayern“ (1: Bayern, 0: sonst), „Alter_z“ (Alter-50.51127), „Abi“ (Schulabschluss, zwei Kategorien), „PI“ (Parteiidentifikation zugunsten einer bestimmten Partei, sechs Kategorien), „pol_Interesse“ (politisches Interesse, fünf Kategorien), „Income“ (Haushaltseinkommen, metrisch), „Demozufrieden“ (Zufriedenheit mit der Demokratie, fünf Kategorien), „Religion“ (neun Kategorien), „Beruf“ (acht Kategorien) und „Arbeitslosigkeit“ (1: z.Zt. arbeitslos, 0: sonst). Die Kovariable „Bayern“ wurde für die Analysen, außer bei der Bestimmung des „passenden“ Scores, nicht berücksichtigt. Eine genauere Darstellung der Variablen ist im Dokument „VariablenDef BTW09small.docx“ auf der beigefügten CD zu finden. Für das zur Verfügung Stellen des Datensatzes und der Datei „VariablenDef BTW09small.docx“ sei Herrn Prof. Dr. Paul W. Thurner (Geschwister-Scholl-Institut für Politikwissenschaft, Lehrstuhl für Empirische Politikforschung und Policy Analysis, LMU München) an dieser Stelle herzlich gedankt.

Bei der Anwendung des Algorithmus auf den Datensatz tritt jedoch folgendes Problem auf. Der Algorithmus, wie er in der Funktion `plregr` implementiert ist,

kann nur Schätzungen für Datensätze ohne fehlende Werte berechnen. Hiervon waren jedoch einige im hier verwendeten Datensatz enthalten. Jede Beobachtung, welche mindestens einen fehlenden Wert aufweist, hätte entfernt werden müssen. Bei einem solchen Vorgehen wären von den ursprünglich 2173 im Datensatz enthaltenen Beobachtungen lediglich 249 übriggeblieben. Bei der Anwendung des Algorithmus auf den dabei entstandenen Datensatz kam es zu Schätzproblemen. Daher wurde eine alternative Herangehensweise gewählt.

Zunächst wurden alle Beobachtungen, welche einen fehlenden Wert in der Zielvariable besitzen, entfernt. Danach waren noch 1428 Beobachtungen im Datensatz vorhanden. Anschließend wurden die Variablen bestimmt, welche besonders viele fehlende Werte beinhalten. Dies waren die Kovariablen „Beruf“ mit 744 fehlenden Werte, „Income“ mit 467 fehlenden Werten und „PI“ mit 287 fehlenden Werten. Nach Entfernung der Beobachtungen mit einem fehlenden Wert in der Kovariable „Beruf“ waren bei der Variable „Income“ noch 225 und nach Entfernung der Beobachtungen mit fehlendem Wert bei „Beruf“ und/oder „Income“ noch 106 fehlende Werte bei der Variable „PI“ enthalten. Um nicht all diese Beobachtungen aus dem Datensatz nehmen zu müssen, wurden diese drei Kovariablen aus dem Datensatz entfernt und danach die Beobachtungen mit mindestens einem fehlenden Wert bei den übrigen globalen Einflussgrößen und/oder in den Variablen zu den Differenzen, also den kategorienspezifischen Einflussgrößen, entfernt. Mit dieser Vorgehensweise waren im Datensatz 816 Beobachtungen mit 9 globalen und 4 kategorienspezifischen Einflussgrößen vorhanden.

Zudem wurden die Variablen „pol_Interesse“ und „Demozufrieden“ dichotomisiert und die Variable „Religion“ trichotomisiert, da nach der Durchführung des oben beschriebenen Vorgehens teilweise nur noch recht wenige Beobachtungen in den ursprünglichen Kategorien vorhanden waren. Bei den ersten beiden Kovariablen wurden die jeweils ersten beiden Kategorien („sehr starkes politisches Interesse“ und „ziemlich stark“ bzw. „sehr zufrieden mit der Demokratie“ und „ziemlich zufrieden“) und die restlichen zu jeweils einer Kategorie zusammengefasst. Bei der Variable „Religion“ wurde die erste Kategorie („evangelische Kirche (ohne Freikirchen)“) unverändert übernommen. Als zweite diente die im ursprünglichen Datensatz dritte Kategorie („römisch-katholische Kirche“), während alle übrigen Ausprägungen zusammengefasst und als dritte Kategorie verwendet wurden.

Im Datensatz war außerdem eine Variable „ldf_nr“, zu welcher im Dokument „VariablenDef BTW09small.docx“ keine Angabe gemacht wurde. Aufgrund der Annahme, dass es sich hierbei um eine Identifikationsnummer der jeweiligen Beochbachtung handelt, wurde sie bei den Analysen nicht berücksichtigt.

Auch hier wurde der „optimale“ λ-Wert über eine Kreuzvalidierung, wie in Kapitel 5.1 beschrieben, mit dem Brier Score als Kriterium ermittelt. Es wurden die Einstellungen `maxiter.opt` $= 10$, `maxiter` $= 10$, $\epsilon_{\text{opt}} = 10^{-5}$ und $\epsilon = 10^{-5}$ für die Berechnungen gewählt. `maxiter.a` wurde bei der Berechnung der Kreuzvalidierung und des endgültigen Modells auf 200 festgelegt. Zur Bestimmung der λ-Sequenz wurde die Funktion „`plregr.seq`“ (s. die beigefügte CD), über welche

auch Koeffizientenpfade berechenbar sind, verwendet. Dabei wurden die in dieser Funktion analog verwendeten Einstellungen bzgl. der maximalen Anzahl an Optimierungsschritten innerhalb von `optim` und dem Abbruchkriterium über ein ϵ ebenfalls auf `maxiter` $= 10$ und `epsilon` $= 10^{-5}$ gesetzt. Die Einstellungen bzgl. des Parameters `maxiter.a` wurden bei der Funktion `plregr.seq` ebenfalls auf 200 festgelegt. Über `maxiter.a` in den Funktionen `plregr.cv` und `plregr.seq` wird jeweils der gleichnamige Parameter in der Funktion `plregr`, welche innerhalb von `plregr.cv` und `plregr.seq` aufgerufen werden, angesteuert. Anhand der Datei „RDAnalyse.r“ auf der beigefügten CD können Analysen, analog zu den hier gezeigten, durchgeführt werden. Bei der Analyse des hier beschriebenen Datensatzes wurde das `R`-Paket „`foreign`“ (R-Core Members et al., 2010) verwendet.

Ergebnisse: Aus den Abbildungen 19 und 20 ist zu erkennen, dass bei der Berechnung der der Koeffizienten entlang der λ-Sequenz selbst beim niedrigsten λ-Wert einige Kovariablen nicht auf einen von Null verschiedenen Wert geschätzt wurden. Um die Pfadverläufe bzgl. der einzelnen Einflussgrößen bei den unteren λ-Werten besser erkennen zu können, wurde in Abbildung 21 die x-Achse, an welcher die λ-Werte abgetragen sind, logarithmiert. Im Endmodell waren lediglich die Variablen „West“, „Alter_z“, „Gewerk“, „Demozufrieden“ und „Religion“, sowie die Scoredifferenz bzgl. „Links-Rechts-Einstufung“, als einzige kategorienspezifische Einflussgröße (s. Abbildung 22), enthalten.

Für den Intercept und die globalen Einflussgrößen, deren Einfluss auf einen von Null verschiedenen Wert geschätzt wurden, ergab sich die geschätzte Koeffizientenmatrix mit gerundeten Werten

$$\hat{\boldsymbol{\beta}}^{\star} = \begin{pmatrix} -0.2961 & 0.4430 - 0.00210.4316 & 0.2650 & -0.3589 & 0.0680 \\ -1.0491 & 0.0315 - 0.01770.1247 & 0.4219 & -0.3214 & 0.5055 \\ -1.3083 & 0.4859 - 0.03270.0660 & 0.4066 & -0.4396 & 0.4327 \\ -0.3288 & 0.0728 - 0.01170.7050 & 0.8900 & -0.5617 & 0.2532 \end{pmatrix}$$

in der Reihenfolge Intercept, „West“, „Alter_z“, „Gewerk“, „Demozufrieden“ und „Religion_2“ und „Religion_3“ (von links nach rechts). Mit „Religion_2“ bzw. „Religion_3“ ist der jeweilige Koeffizientenvektor zur zweiten („römisch-katholische Kirche“) bzw. dritten Kategorie (übrige Ausprägungen) der Variable „Religion“ gemeint. Die erste Kategorie („evangelische Kirche (ohne Freikirchen)“) war hierbei die Referenzkategorie. Die Koeffizienten zu den übrigen Variablen wurden vom Algorithmus auf Null geschätzt. Für $\hat{\gamma}_{\text{LiRe}}$, als geschätzter gerundeter Koeffizient für die Scoredifferenzen bzgl. „Links-Rechts-Einstufung“, ergab sich

$$\hat{\gamma}_{\text{LiRe}} = -0.6863.$$

Auffallend ist, dass die Effekte zu den einzelnen Kovariablen über die Referenzkategorien hinweg stets dasselbe Vorzeichen besitzen. So wirkt sich bspw. die

Tatsache, dass der Befragte aus Westdeutschland stammt, positiv auf die einzelnen Wahrscheinlichkeiten für eine Wahlabsicht zugunsten einer anderen Partei als der CDU/CSU aus. Die Koeffizienten zur Variable „Alter_z“ besitzen alle ein negatives Vorzeichen, wobei darauf zu achten ist, dass diese Variable metrisch in das Modell aufgenommen wurde. Hier war der geschätzte Effekt jedoch eher gering. Die Tatsache, Gewerkschaftsmitglied zu sein, wirkt sich zusammenfassend negativ auf die Wahrscheinlichkeit für eine Wahlabsicht zugunsten der CDU/CSU aus. Ebenso verhält sich der geschätzte Effekt bzgl. Personen, welche sehr oder ziemlich zufrieden mit der Demokratie sind. Die geschätzten Einflüsse der Religionszugehörigkeit besitzen pro Kategorie ebenfalls dasselbe Vorzeichen.

Die kategorienspezifische Einflussgröße „Links-Rechts-Einstufung“ weißt einen negativen geschätzten Effekt auf, was bedeutet, dass eine große Differenz zwischen dem Score des Befragten und der jeweiligen Partei sich negativ auf die Wahrscheinlichkeit für eine Wahlabsicht zugunsten der entsprechenden Partei auswirkt.

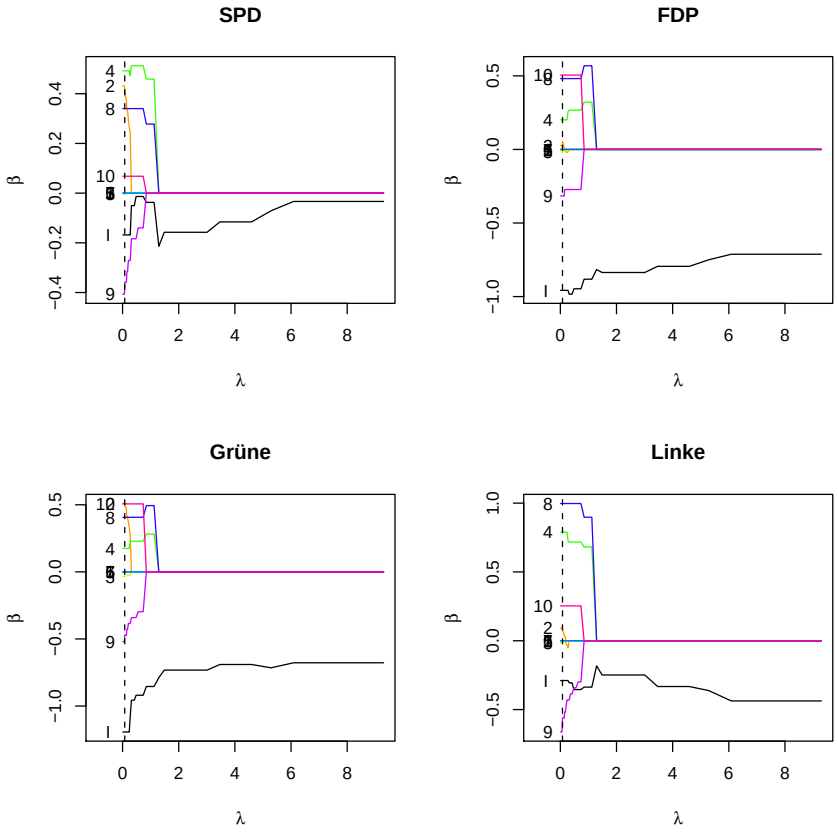

Abbildung 19: β-Pfade zum realen Datensatz aufgeteilt nach den Kategorien der Zielvariable; Die Pfade sind nach den Koeffizientenvektoren beschriftet: I: Intercept, 1: „Geschlecht“, 2: „West“, 3: „Alter_z“, 4: „Gewerk“, 5: „Abi“, 6: „Arbeitslos“, 7: „pol_Interesse“, 8: „Demozufrieden“, 9: Koeffizientenvektor für die zweite Kategorie von Religion („römisch-katholische Kirche“), 10: Koeffizientenvektor die für dritte Kategorie von Religion

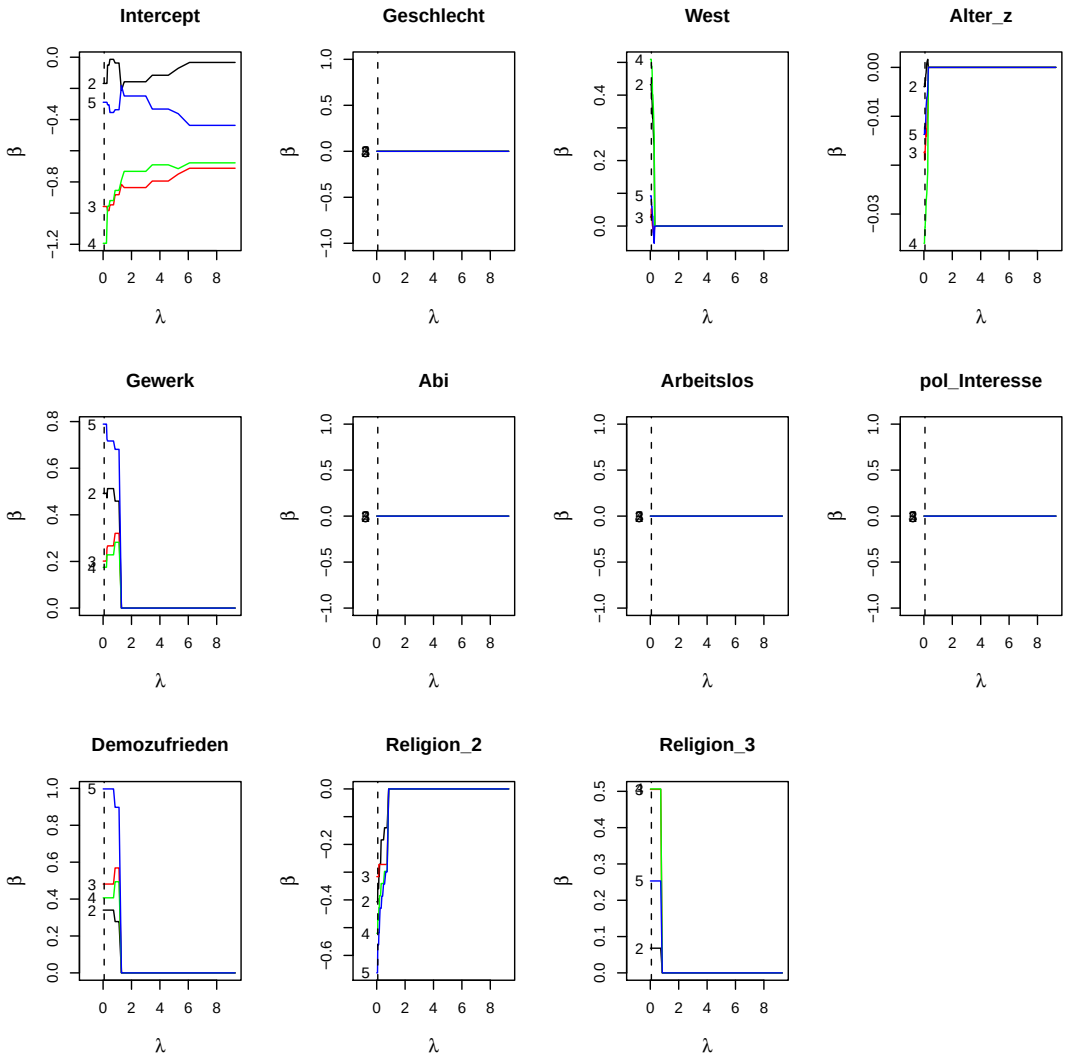

Abbildung 20: β-Pfade zum realen Datensatz aufgeteilt nach Kovariablen; Die Pfade sind nach den Referenzkategorien beschriftet: 2: SPD, 3: FDP, 4: Grüne, 5: Linke

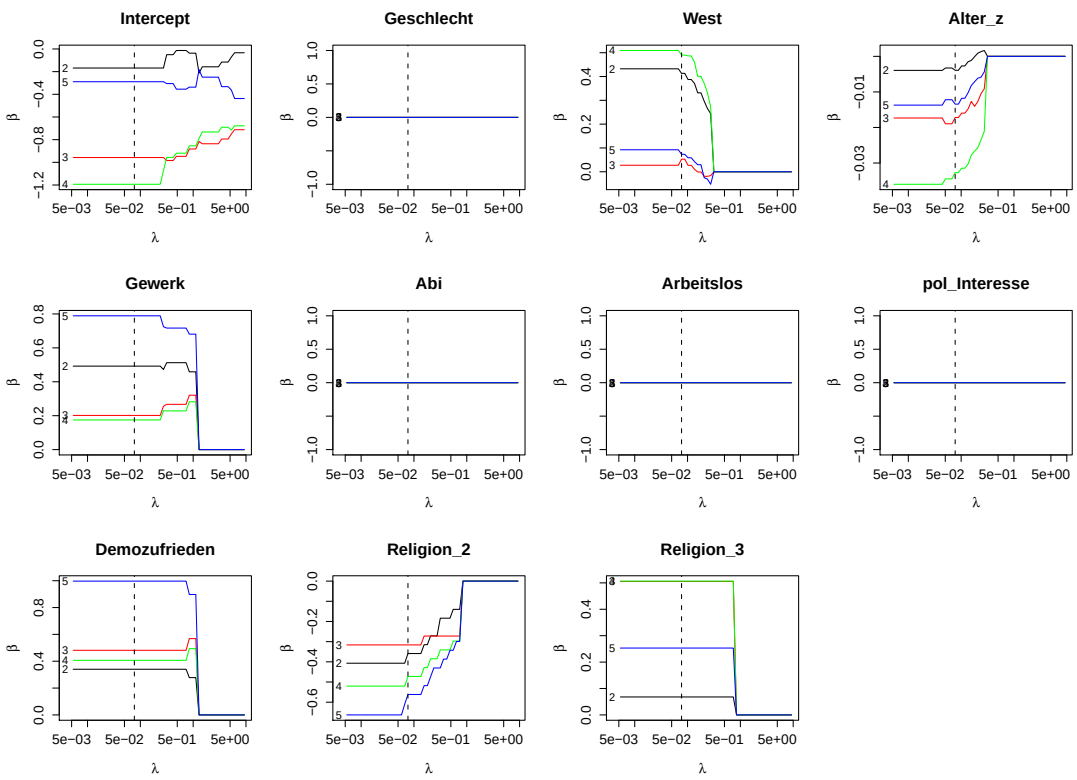

Abbildung 21: β-Pfade mit logarithmierter x-Achse zum realen Datensatz aufgeteilt nach Kovariablen; Die Pfade sind nach den Referenzkategorien beschriftet: 2: SPD, 3: FDP, 4: Grüne, 5: Linke

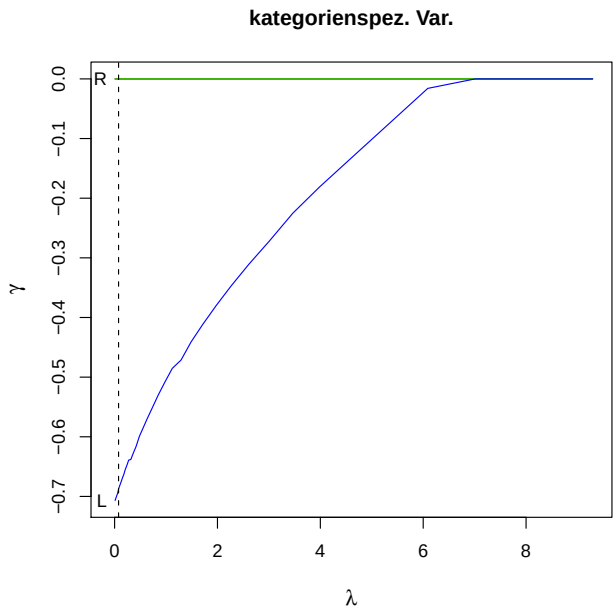

Abbildung 22: γ-Pfade zum realen Datensatz; Die Pfade sind nach den kategorienspezifischen Einflussgrößen, den Scoredifferenzen, beschriftet: „L“ steht für „Links-Rechts-Einstufung“, „R“ (Rest) für die drei Einflussgrößen „sozioökonomische Dimension“, „Zuzug von Ausländern“, „Kernenergie“,

6 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Algorithmus für kategoriale Responsemodelle vorgestellt. Dieser ist in der Lage, im Gegensatz zu Verfahren, die Parameter getrennt voneinander penalisieren, über Regularisierungstechniken gesamte Kovariablen zu selektieren. Dabei findet ein Strafterm aus dem Lasso-Kontext Anwendung. Neben metrischen Einflussgrößen können auch kategoriale, welche wiederum, entsprechend dem Group Lasso, insgesamt berücksichtigt oder aus dem Modell genommen werden, und kategorienspezifische im behandelten Datensatz vorhanden sein. Es ist direkte Variablenselektion möglich, wodurch die Modellkomplexität verringert werden kann, während die im Modell verbleibenden Koeffizienten, bei einem entsprechenden Wert des Penalisierungsparameters λ, geshrinkt werden. Wie in der Simulationsstudie gezeigt, wirken sich diese beiden Eigenschaften des Algorithmus in vielen Datensituationen bzgl. des MSE($\boldsymbol{\pi}$) und MSE($\boldsymbol{\beta}$) positiv gegenüber dem ML-Schätzer auf die Schätzungen aus.

Eine Problematik jedoch, die sich beim Testen des Algorithmus bspw. in Simulationen ergab, stellt die Rechenzeit dar. In der hier verwendeten Form liegen diese sehr hoch.

Prinzipiell ist der Algorithmus auch im $p > n$-Fall anwendbar. Bei Tests der hierzu entwickelten R-Funktionen kam es diesbezüglich jedoch zu Problemen (getestet wurden die entsprechenden Funktionen auf der beigefügten CD).

Eine direkte Erweiterung des Algorithmus wäre, anstatt mit einer Referenzkategorie, mit symmetrischer Nebenbedingung zu arbeiten. In Zahid und Tutz (2009) wird die Verwendung einer symmetrischen Nebenbedingung als „more appropriate“ (Zahid und Tutz, 2009, S. 3) bzw. „more natural“ (Zahid und Tutz, 2009, S. 8) in Verbindung mit Regularisierungsverfahren bezeichnet, was damit zu begründen wäre, dass die Parameter dadurch gegen einen Mittelwert und nicht gegen eine Referenzkategorie, wie in dem hier vorgestellten Algorithmus, geshrinkt werden. Jedoch sollten die entsprechenden Ergebnisse inhaltlich nicht stark von den hier gezeigten abweichen. Zudem sollten die Schätzungen bei Verwendung einer Referenzkategorie, wie in Kapitel 2.2 (bzw. besonders in Kapitel 2.2.1) deutlich werden sollte, leichter, als diejenigen, welche man aus einem Modell mit symmetrischer Nebenbedingung erhält, interpretierbar sein.

Literatur

[1] Bates, D. ; Maechler, M.: *Matrix: Sparse and Dense Matrix Classes and Methods*, 2010. http://CRAN.R-project.org/package=Matrix. – R package version 0.999375-40

[2] Brier, G. W.: Verification of Forecasts Expressed in Terms of Probability. In: *Monthly Weather Review* 78 (1950), Nr. 1, S. 1–3

[3] Croissant, Y.: *mlogit: multinomial logit model*, 2010. http://CRAN.R-project.org/package=mlogit. – R package version 0.1-8

[4] Fahrmeir, L. ; Kneib, T. ; Lang, S.: *Regression - Modelle, Methoden und Anwendungen*. Berlin : Springer, 2007. – ISBN 978-3-540-33932-8

[5] Friedman, J. ; Hastie, T. ; Tibshirani, R.: Regularization Paths for Generalized Linear Models via Coordinate Descent. In: *Journal of Statistical Software* 33 (2010), Nr. 1, S. 1–22

[6] Hastie, T. ; Tibshirani, R. ; Friedman, J.: *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. 2. Auflage. New York : Springer, 2009. – ISBN 978-0-387-84857-0

[7] Hoerl, A. E. ; Kennard, R. W.: Ridge Regression: Biased Estimation for Nonorthogonal Problems. In: *Technometrics* 12 (1970), Nr. 1, S. 55–67

[8] Meier, L.: *grplasso: Fitting user specified models with Group Lasso penalty*, 2009. http://CRAN.R-project.org/package=grplasso. – R package version 0.4-2

[9] Meier, L. ; van de Geer, S. ; Bühlmann, P.: The group lasso for logistic regression. In: *Journal of the Royal Statistical Society, Series B* 70 (2008), Nr. 1, S. 53–71

[10] R-core members ; DebRoy, S. ; Bivand, R. ; others: see COPYRIGHTS file in the sources.: *foreign: Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ...*, 2010. http://CRAN.R-project.org/package=foreign. – R package version 0.8-40

[11] R Development Core Team: *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2010. http://www.R-project.org. – ISBN 3-900051-07-0

[12] Sarkar, D.: *lattice: Lattice Graphics*, 2010. http://CRAN.R-project.org/package=lattice. – R package version 0.18-8

[13] Schaback, R. ; Wendland, H.: *Numerische Mathematik*. 5., vollständig neu bearbeitete Auflage. Berlin : Springer, 2005. – ISBN 3-540-21394-5

[14] Smyth, Y. G. with contributions from H. G. with contributions from Hu ; Dunn, P. ; Phipson, B.: *statmod: Statistical Modeling*, 2010. `http://CRAN.R-project.org/package=statmod`. – R package version 1.4.8

[15] Thurner, P. W. ; Eymann, A.: Policy-specific alienation and indifference in the calculus of voting: A simultaneous model of party choice and abstention. In: *Public Choice* 102 (2000), Nr. 1/2, S. 51–77

[16] Tibshirani, R.: Regression Shrinkage and Selection via the Lasso. In: *Journal of the Royal Statistical Society. Series B* 58 (1996), Nr. 1, S. 267–288

[17] Tutz, G.: *Structured Regression for Categorical Data*. Cambridge : Cambridge University Press, 2011

[18] Venables, W. N. ; Ripley, B. D.: *Modern Applied Statistics with S*. 4. Auflage. New York : Springer, 2002 `http://www.stats.ox.ac.uk/pub/MASS4`. – ISBN 0-387-95457-0

[19] Yuan, M. ; Lin, Y.: Model selection and estimation in regression with grouped variables. In: *Journal of the Royal Statistical Society, Series B* 68 (2006), Nr. 1, S. 49–67

[20] Zahid, F. M. ; Tutz, G.: Ridge Estimation for Multinomial Logit Models with Symmetric Side Constraints / Ludwig-Maximilians-Universität München, Institut für Statistik. 2009. – Technical Report Nr. 067

[21] Zahid, F. M. ; Tutz, G.: Multinomial Logit Models with Implicit Variable Selection / Ludwig-Maximilians-Universität München, Institut für Statistik. 2010. – Technical Report Nr. 089

[22] Zeileis, A. ; Croissant, Y.: Extended Model Formulas in R: Multiple Parts and Multiple Responses. In: *Journal of Statistical Software* 34 (2010), Nr. 1, 1–13. `http://www.jstatsoft.org/v34/i01/`

[23] Zeileis, A. ; Grothendieck, G.: zoo: S3 Infrastructure for Regular and Irregular Time Series. In: *Journal of Statistical Software* 14 (2005), Nr. 6, 1–27. `http://www.jstatsoft.org/v14/i06/`

[24] Zeileis, A. ; Hothorn, T.: Diagnostic Checking in Regression Relationships. In: *R News* 2 (2002), Nr. 3, 7–10. `http://CRAN.R-project.org/doc/Rnews/`

[25] Zou, H. ; Hastie, T.: Regularization and variable selection via the elastic net. In: *Journal of the Royal Statistical Society, Series B* 67 (2005), Nr. 2, S. 301–320

Anhang A

In diesem Anhang ist eine R-Funktion zum im Kapitel 4.6 vorgestellten Algorithmus zur Merkmalsauswahl in kategorialen Responsemodellen aufgeführt (vgl. Kapitel 4). In dem Dokument „Algorithmus.r“ auf der CD zu dieser Arbeit befindet sich eine kommentierte und optisch etwas andere Version dieser Funktion.

Die unten aufgeführte Funktion besitzt folgende Argumente:

- `xy`: ein `data.frame` der Dimension $n \times (G+1)$; Die erste Spalte ist dabei der Responsevektor. Hierfür kann aus zwei Kodierungsmöglichkeiten gewählt werden:

  1. als factor-Variable; Referenzkategorie ist hierbei die Kategorie, welche nach Anwendung von „as.numeric“ auf den entsprechenden Vektor mit 1 kodiert ist.
  2. als numeric-Variable mit den Ausprägungen 1, 2, ... (natürliche Zahlen von 1 bis K, mit K als die Anzahl der Responsekategorien, ohne eine Zahl in der Folge auszulassen); Referenzkategorie ist hierbei die Kategorie, welche mit 1 kodiert ist. Rest von xy (sprich `xy[, -1]` in R): (mindestens zwei) Prädiktoren (auch factor-Variablen)

- `x.kat`: eine `matrix` der Dimension $n \cdot K \times L$; Da die erste Responsekategorie als Referenzkategorie gewählt wird, entsprechen dieser die erste, die $(k+1)$-te, $(k+2)$-te, ... Zeile; optional

- `xneu`: ein `data.frame` oder eine `matrix` mit neuen Daten entsprechend zu `x` (optional); Daten auf die der aus der Funktion gewonnene Fit angewendet wird; optional

- `x.katneu`: eine `matrix` mit neuen Daten entsprechend zu `x.kat`; s. `xneu`; kann nur verwendet werden, wenn an `xneu` ebenfalls Daten übergeben werden; optional

- `yneu`: zu `xneu` und ggf. `x.katneu` gehörender Response-`vector`; optional

- `lambda`: λ-Wert für die Penalisierung; Je höher dieser Wert ist, desto „stärker“ ist die Penalisierung

- `epsilon`: das ϵ aus dem Konvergenzkriterium

- `be.start`: eine Matrix der Dimension $(K-1) \times (p+1)$ mit Startwerten für die Koeffizienten; Die erste Responsekategorie dient hierbei als Referenzkategorie. Falls `y` binär ist, ist hierfür entsprechend ein Vektor zu übergeben; optional

- `ga.start`: Startwertvektor für γ; Falls `beta.start` spezifiziert wird, jedoch `ga.start` nicht, so werden für γ Werte aus einer $N(0,1)$-Verteilung gezogen; Falls ga.start spezifiziert wird, muss auch `sd.opt` angegeben werden; optional

- `mult`: logisches Argument (standardmäßig auf `F` gesetzt); Falls `mult = T` eingestellt ist, so wird für die Startwertberechnung die Funktion `multinom` aus dem Paket „`nnet`“ (Venables und Ripley, 2002) verwendet. Startwerte werden nur berechnet, falls keine Startwerte vorgegeben werden (s. `be.start`).

- `mlog`: logisches Argument (standardmäßig auf `T` gesetzt); Falls `mlog = T` gesetzt ist, wird die Startwertberechnung über die Funktion `mlogit` aus dem gleichnamigen Paket „`mlogit`“ (Croissant, 2010) durchgeführt. Startwerte für $\boldsymbol{\gamma}$ können hier nur über diese Funktion berechnet werden. Falls `mult = T` gesetzt ist, wird `mlog = F` gesetzt. Falls diese Einstellungen gewählt werden, werden für $\boldsymbol{\gamma}$ ggf. zufällige Werte aus einer (univariaten) $N(0,1)$-Verteilung gezogen. Startwerte werden nur berechnet, falls keine Startwerte für $\boldsymbol{\beta}$ vorgegeben werden (s. `ga.start`).

- `maxiter`: Anzahl der maximalen Iterationen bei der Optimierung der einzelnen Koeffizienten innerhalb der Funktin „`optim`“ (s. `control` bzw. `maxit` in der Funktion `optim` in `R`; Die Wahl dieses Parameters, wie auch von `epsilon`, kann Einfluss auf die Geschwindigkeit des Algorithmus haben.

- `maxiter.a`: maximale Anzahl an Iterationen (= `maxiter.a` + 1) bei der Optimierung (maximale Anzahl an Durchläufen durch alle Kovariablen)

- sd.opt: Vektor der Länge von ncol(x.kat); zur Beeinflussung der Intervalle, in welchen nach dem optimalen Werten für gamma gesucht wird (s.u. bei der Verwendung von „`optimize`“); optional

Entsprechend werden je nach Wahl der Argumente folgende Rückgabewerte ausgegeben:

- `koeff`: β-Werte aus der Schätzung; Die Dimension entspricht der von `be.start`. Man beachte, dass bei kategorialen Kovariablen die erste Kategorie als Referenzkategorie verwendet wird.

- `loglik.pen`: Wert der negativen penalisierten Log-Likelihood

- `lambda`: verwendeter λ-Wert

- `ga`: Schätzungen für γ; Die Dimension entspricht der von `ga.start`.

- `pi.fit`: gefittete Wahrscheinlichkeiten für `yneu` (unter Verwendung von `be`)

- `y.fit`: gefittete Responsewerte für `xneu`, falls dieses spezifiziert wurde
- `missklass`: Anzahl der Fehlklassifikationen, falls `xneu` und `yneu` spezifiziert wurden
- `loglik.pr`: prädiktive negative penalisierte Log-Likelihood für neue Daten
- `brier`: prädiktiver Brier Score für neue Daten
- `fit.unpen`: Fit aus unpenalisierter Schätzung über Funktion `multinom` bzw. `mlogit`; Wird nur berechnet und ausgegeben, falls `be.start` nicht spezifiziert wurde.
- `sd.ga` s. oben `sd.opt`; Dieser Vektor wird bei fehlender Spezifikation von `be.start` innerhalb der Funktion `plregr` als Wurzel der negativen Diagonalelemente der Hesse-Matrix, welche über die Funktion `mlogit` aus dem gleichnamigen Paket berechnet wird, bestimmt. Hierfür muss `mlog = T` gesetzt werden. Falls sd.opt spezifiert wurde, wird der entsprechende Vektor ausgegeben.

In dieser Funktion werden die Pakete `nnet` und `mlogit` mit den Paketen „`Formula`“ (Zeileis und Croissant, 2010), „`statmod`“ (Smyth et al., 2010), „`lmtest`“ (Zeileis und Hothorn, 2002) sowie „`zoo`“ (Zeileis und Grothendieck, 2005) verwendet. Außerdem fand noch das Paket „`Matrix`“ (Bates und Maechler, 2010) mit dem Paket „`lattice`“ (Sarkar, 2010) Verwendung.

```
> plregr <- function(xy, x.kat, xneu, x.katneu, yneu, lambda, epsilon = 1e-05,
+     be.start, ga.start, mult = F, mlog = T, maxiter = 10, maxiter.a = 100,
+     sd.opt, ...) {
+     if (mult)
+         mlog <- F
+     x.k.l <- F
+     y <- as.numeric(xy[, 1])
+     x <- x.or <- xy[, -1]
+     if (!missing(x.kat)) {
+         x.kat.mlog <- x.kat
+         x.kat.ref <- c()
+         for (i in 0:(nrow(x.kat)/max(y) - 1)) x.kat.ref <- c(x.kat.ref,
+             i * max(y) + 1)
+         for (i in x.kat.ref) x.kat[i:(i + max(y) - 1), ] <- t(t(x.kat[i:(i +
+             max(y) - 1), ]) - x.kat[i, ])
+         x.kat <- x.kat[-x.kat.ref, ]
+         x.k.l <- T
+     }
+     umw <- function(data) {
+         data.mat <- matrix(1, nrow = nrow(data), ncol = 0)
+         j.list <- list(1)
+         ind <- 2
+         for (j in 1:ncol(data)) {
+             if (is.factor(data[, j])) {
+                 mx.j <- max(as.numeric(data[, j])) - 1
+                 j.list[[j + 1]] <- ind:(ind + mx.j - 1)
+                 ind <- ind + mx.j
+                 j.mat <- matrix(0, ncol = mx.j, nrow = nrow(data))
+                 for (i in 1:nrow(data)) {
```

```
+                   j.mat[i, as.numeric(data[i, j]) - 1] <- 1
+                 }
+                 for (i in 1:nrow(data)) if (as.numeric(data[i,
+                   j]) == 1)
+                   j.mat[i, ] <- -1
+                 data.mat <- cbind(data.mat, j.mat)
+             }
+             else {
+                 j.list[[j + 1]] <- ind
+                 ind <- ind + 1
+                 data.mat <- cbind(data.mat, data[, j])
+             }
+         }
+         return(list(data.mat = data.mat, j.list = j.list))
+     }
+     umw.x <- umw(x)
+     x <- umw.x$data.mat
+     jl <- umw.x$j.list
+     jl.l <- length(jl)
+     if (!missing(x.kat))
+         for (i in 1:ncol(x.kat)) jl[[i + jl.l]] <- max(jl[[jl.l]]) +
+             i
+     if (missing(be.start)) {
+         if (mult) {
+             require(nnet)
+             x.data <- as.data.frame(cbind(x, y))
+             form <- as.formula(paste("y~", paste(names(x.data)[1:ncol(x)],
+                 collapse = " + ", sep = ""), sep = ""))
+             fit.mult <- multinom(form, data = x.data)
+             be <- coef(fit.mult)
+         }
+         else {
+             if (mlog) {
+                 require(mlogit)
+                 y.mlog <- rep(0, max(y) * nrow(x))
+                 for (i in 0:length(y)) y.mlog[i * max(y) + y[i +
+                   1]] <- 1
+                 altern <- rep(1:max(y), length(y))
+                 x.mlog <- matrix(0, nrow = max(y) * nrow(x),
+                   ncol = ncol(x))
+                 for (i in 1:ncol(x)) x.mlog[, i] <- rep(x[, i],
+                   each = max(y))
+                 data.mlog.a <- as.data.frame(cbind(y.mlog, altern,
+                   if (!missing(x.kat))
+                     x.kat.mlog, x.mlog))
+                 data.mlog <- mlogit.data(data.mlog.a, choice = "y.mlog",
+                   shape = "long", alt.var = "altern")
+                 if (!missing(x.kat)) {
+                   nam.katsp <- paste(names(data.mlog.a)[3:(ncol(x.kat.mlog) +
+                     2)], collapse = "+", sep = "")
+                   nam.glob <- paste(names(data.mlog.a)[(ncol(x.kat.mlog) +
+                     3):(ncol(x.kat.mlog) + ncol(x.mlog) + 2)],
+                     collapse = "+", sep = "")
+                   form.mlog <- as.formula(paste(paste("y.mlog ~",
+                     nam.katsp, sep = ""), nam.glob, sep = "|"))
+                 }
+                 else {
+                   nam.glob <- paste(names(data.mlog.a)[3:(ncol(x.mlog) +
+                     2)], collapse = "+", sep = "")
+                   form.mlog <- as.formula(paste(paste("y.mlog ~",
+                     0, sep = ""), nam.glob, sep = "|"))
+                 }
+                 fit.mult <- mlogit(form.mlog, data = data.mlog)
+                 if (!missing(x.kat)) {
+                   be <- matrix(fit.mult$coefficients[-(max(y):(ncol(x.kat.mlog) +
```

```
+                   max(y) - 1))], nrow = max(y) - 1)
+                 ga <- fit.mult$coefficients[max(y):(ncol(x.kat.mlog) +
+                   max(y) - 1)]
+                 sd.ga <- sqrt(-diag(fit.mult$hessian)^-1)[max(y):(ncol(x.kat.mlog) +
+                   max(y) - 1)]
+               }
+               else {
+                 be <- matrix(fit.mult$coefficients, nrow = max(y) -
+                   1)
+                 ga <- 0
+               }
+           }
+           else be <- matrix(rnorm((ncol(x) + 1) * (max(y) -
+               1)), ncol = ncol(x) + 1, nrow = max(y) - 1)
+       }
+   }
+   else {
+       be <- be.start
+       mult <- mlog <- F
+   }
+   if (!missing(x.kat)) {
+       if (missing(ga.start)) {
+           if (!mlog)
+               ga <- rnorm(ncol(x.kat))
+       }
+       else {
+           ga <- ga.start
+       }
+   }
+   else ga <- 0
+   if (!missing(sd.opt))
+       sd.ga <- sd.opt
+   if (max(y) == 2 & mult)
+       be <- t(as.matrix(be))
+   if (max(y) == 2 & !missing(be.start))
+       be <- t(as.matrix(be))
+   xqr <- matrix(0, nrow = nrow(x), ncol = ncol(x))
+   for (j in 2:jl.l) xqr[, (jl[[j]] - 1)] <- qr.Q(qr(x[, (jl[[j]] -
+       1)]))
+   xd <- cbind(1, xqr)
+   for (j in 2:jl.l) {
+       if (length(jl[[j]]) > 1 & max(y) > 2) {
+           be[, jl[[j]]] <- t(tcrossprod(qr.R(qr(x[, (jl[[j]] -
+               1)])), be[, jl[[j]]]))
+       }
+       else {
+           be[, jl[[j]]] <- t(qr.R(qr(x[, (jl[[j]] - 1)])) %*%
+               be[, jl[[j]]])
+       }
+   }
+   y.mat <- matrix(0, nrow = length(y), ncol = max(y))
+   for (i in 1:length(y)) {
+       y.mat[i, y[i]] <- 1
+   }
+   if (!missing(x.kat)) {
+       y.list <- list()
+       y.mat.kat <- y.mat[, -1]
+       for (i in 1:length(y)) y.list[[i]] <- if (is.matrix(y.mat.kat))
+           t(y.mat.kat[i, ])
+       else y.mat.kat[i]
+       xkatd <- matrix(0, nrow = nrow(x.kat), ncol = ncol(x.kat))
+       for (j in 1:ncol(x.kat)) xkatd[, j] <- qr.Q(qr(x.kat[,
+           j]))
+       for (j in 1:ncol(x.kat)) ga[j] <- qr.R(qr(x.kat[, j])) %*%
+           ga[j]
```

```
+         require(Matrix)
+         y.mat.kat <- bdiag(y.list)
+         Om <- y.mat.kat %*% xkatd
+     }
+     else x.kat.dec <- xkatd <- y.mat.kat <- Om <- 0
+     likeli.j <- function(be.j) {
+         if (max(l) <= ncol(xd)) {
+             logl <- -sum(diag(tcrossprod(tcrossprod(xd[, -l],
+                 benu[, -l]), y.mat))) - sum(diag(tcrossprod(tcrossprod(xd[,
+                 l], rbind(0, matrix(be.j, ncol = length(l)))),
+                 y.mat))) - sum(Om %*% ga.alt) + if (max(y) ==
+                 2)
+                 sum(apply(tcrossprod(xd[, -l], matrix(be.alt[,
+                   -l], ncol = ncol(xd) - length(l))) + if (length(l) >
+                   1) xd[, l] %*% be.j else xd[, l] * be.j + if (x.k.l) matrix(xkatd %*%
+                   ga.alt, ncol = max(y) - 1, byrow = T) else 0,
+                   1, function(x) log(1 + sum(exp(x)))))
+             else sum(apply(tcrossprod(xd[, -l], be.alt[, -l]) +
+                 tcrossprod(xd[, l], matrix(be.j, ncol = length(l))) +
+                 if (x.k.l) matrix(xkatd %*% ga.alt, ncol = max(y) -
+                   1, byrow = T) else 0, 1, function(x) log(1 +
+                 sum(exp(x)))))
+             pen <- 0
+             for (z in 1:length(jl)) if (jl[[z]][1] != 1 & l[1] !=
+                 jl[[z]][1] & max(jl[[z]]) <= ncol(xd))
+                 pen <- pen + sqrt(sum(be.alt[, jl[[z]]]^2)) *
+                   sqrt(length(jl[[z]]))
+             pen <- pen * sqrt(max(y) - 1)
+             pen <- pen + sum(abs(ga.alt))
+             if (l[1] != 1)
+                 pen <- pen + sqrt(max(y) - 1) * sqrt(sum(be.j^2))
+             logl <- logl + lambda * pen
+         }
+         else {
+             l <- l - ncol(xd)
+             logl <- -sum(diag(tcrossprod(tcrossprod(xd, benu),
+                 y.mat))) - sum(Om[, -l] %*% ga.alt[-l]) - sum(Om[,
+                 l] * be.j) + sum(apply(tcrossprod(xd, be.alt) +
+                 matrix(xkatd[, -l] %*% ga.alt[-l], ncol = max(y) -
+                   1, byrow = T) + matrix(xkatd[, l] * be.j, ncol = max(y) -
+                 1, byrow = T), 1, function(x) log(1 + sum(exp(x)))))
+             pen <- 0
+             for (z in 1:length(jl)) if (jl[[z]][1] != 1 & max(jl[[z]]) <=
+                 ncol(xd))
+                 pen <- pen + sum(sqrt(be.alt[, jl[[z]]]^2)) *
+                   sqrt(length(jl[[z]]))
+             pen <- pen * sqrt(max(y) - 1)
+             pen <- pen + sum(abs(ga.alt[-l]))
+             pen <- pen + abs(be.j)
+             logl <- logl + lambda * pen
+         }
+         return(logl)
+     }
+     be.alt <- matrix(0, nrow = nrow(be), ncol = ncol(be))
+     ga.alt <- if (!missing(x.kat))
+         rep(0, length(ga))
+     else 0
+     anz.it <- 0
+     while (sqrt(sum((be.alt - be)^2) + sum((ga.alt - ga)^2))/sqrt(sum(be.alt^2) +
+         sum(ga.alt^2)) > epsilon & anz.it <= maxiter.a) {
+         anz.it <- anz.it + 1
+         be.alt <- be
+         ga.alt <- ga
+         sumdfinv <- sumdfinv.ga <- 0
+         for (k in 2:length(jl)) {
```

```
+             l <- jl[[k]]
+             if (max(l) <= ncol(xd))
+                 sumdfinv <- sumdfinv + length(l)^(-1) * sum(abs(be[,
+                   l]))
+             else {
+                 l <- l - ncol(xd)
+                 sumdfinv.ga <- sumdfinv.ga + abs(ga[l])
+             }
+         }
+         if (sumdfinv == 0)
+             sumdfinv <- Inf
+         if (sumdfinv.ga == 0)
+             sumdfinv.ga <- Inf
+         for (k in 1:length(jl)) {
+             l <- lt <- jl[[k]]
+             if (k == 1) {
+                 benu <- rbind(0, be.alt)
+                 lik.test <- likeli.j(be[, l])
+                 if (lik.test == Inf | lik.test == -Inf | lik.test ==
+                   0) {
+                   be <- be.alt <- matrix(rnorm(nrow(be) * ncol(be)),
+                     nrow = nrow(be), ncol = ncol(be))
+                   benu <- rbind(0, be.alt)
+                   warning("Log-Likelihood gleich Inf, -Inf oder 0",
+                     call. = F)
+                 }
+                 be[, l] <- matrix(optim(be[, l], likeli.j, method = "Nelder-Mead",
+                   control = list(maxit = maxiter))$par, ncol = length(l))
+             }
+             else {
+                 if (max(l) <= ncol(xd)) {
+                   vn.alt <- ((length(l))^(-1) * sum(abs(be[,
+                     l])))/(sumdfinv)
+                   vn <- if (max(y) == 2)
+                     sqrt(sum((crossprod(xd[, l], (y.mat - (cbind(1,
+                       exp(xd[, -l] %*% be.alt[, -l] + if (!missing(x.kat)) matrix(xkatd %*%
+                         ga.alt, ncol = max(y) - 1, byrow = T) else 0))/apply(exp(xd[,
+                       -l] %*% be.alt[, -l] + if (!missing(x.kat)) matrix(xkatd %*%
+                       ga.alt, ncol = max(y) - 1, byrow = T) else 0),
+                       1, function(x) sum(x) + 1)))))^2))
+                   else sqrt(sum((crossprod(xd[, l], (y.mat -
+                     (cbind(1, exp(tcrossprod(xd[, -l], be.alt[,
+                       -l]) + if (!missing(x.kat)) matrix(xkatd %*%
+                       ga.alt, ncol = max(y) - 1, byrow = T) else 0))/apply(exp(tcrossprod(xd[,
+                       -l], be.alt[, -l]) + if (!missing(x.kat)) matrix(xkatd %*%
+                       ga.alt, ncol = max(y) - 1, byrow = T) else 0),
+                       1, function(x) sum(x) + 1)))))^2))
+                 }
+                 else {
+                   l <- max(l) - ncol(xd)
+                   vn.alt <- (abs(ga[l]))/(sumdfinv.ga)
+                   vn <- sqrt(sum((crossprod(matrix(xkatd[, l],
+                     nrow = nrow(x), byrow = T), (y.mat - (cbind(1,
+                     exp(tcrossprod(xd, be.alt) + matrix(xkatd[,
+                       -l] %*% ga.alt[-l], ncol = max(y) - 1,
+                       byrow = T)))/apply(exp(tcrossprod(xd, be.alt) +
+                     matrix(xkatd[, -l] %*% ga.alt[-l], ncol = max(y) -
+                       1, byrow = T)), 1, function(x) sum(x) +
+                     1)))))^2))
+                 }
+                 if (vn.alt == Inf)
+                   vn.alt <- 0
+                 if (max(lt) <= ncol(xd)) {
+                   if (vn <= sqrt(max(y) - 1) * sqrt(length(l)) *
+                     lambda | vn.alt < ((ncol(x))^(-1)))
```

```
+                   be[, l] <- 0
+                 else {
+                   benu <- rbind(0, be.alt)
+                   be[, l] <- matrix(optim(as.vector(be[, l]),
+                     likeli.j, method = "Nelder-Mead", control = list(maxit = maxiter))$par,
+                     ncol = length(l))
+                 }
+               }
+               else {
+                 if (vn <= lambda | vn.alt < ((ncol(x.kat))^(-1)))
+                   ga[l] <- 0
+                 else {
+                   l <- lt
+                   benu <- rbind(0, be.alt)
+                   ga[lt - ncol(xd)] <- optimize(likeli.j, c(-2,
+                     2) * sd.ga[lt - ncol(xd)] + ga.alt[lt -
+                     ncol(xd)])$minimum
+                 }
+               }
+           }
+       }
+   }
+   if (anz.it == maxiter.a + 1)
+       warning(paste(paste("keine Konvergenz nach", maxiter.a),
+           "Iterationen"), call. = F)
+   for (j in 2:jl.l) {
+       if (length(jl[[j]]) > 1 & max(y) > 2) {
+           be[, jl[[j]]] <- t(tcrossprod(solve(qr.R(qr(x[, (jl[[j]] -
+               1)]))), be[, jl[[j]]]))
+       }
+       else {
+           be[, jl[[j]]] <- t(solve(qr.R(qr(x[, (jl[[j]] - 1)]))) %*%
+               be[, jl[[j]]])
+       }
+   }
+   if (!missing(x.kat))
+       for (j in 1:ncol(x.kat)) ga[j] <- solve(qr.R(qr(x.kat[,
+           j]))) %*% ga[j]
+   if (!missing(xneu)) {
+       umw.neu <- function(data) {
+           data.mat <- matrix(1, nrow = nrow(data), ncol = 0)
+           for (j in 1:ncol(data)) {
+               if (is.factor(data[, j])) {
+                 j.mat <- matrix(0, ncol = max(as.numeric(x.or[,
+                   j])) - 1, nrow = nrow(data))
+                 for (i in 1:nrow(data)) {
+                   j.mat[i, as.numeric(data[i, j]) - 1] <- 1
+                 }
+                 data.mat <- cbind(data.mat, j.mat)
+               }
+               else {
+                 data.mat <- cbind(data.mat, data[, j])
+               }
+           }
+           return(data.mat)
+       }
+       xneu <- umw.neu(xneu)
+       xdneu <- cbind(1, xneu)
+       if (!missing(x.katneu)) {
+           x.kat.ref <- c()
+           for (i in 0:(nrow(x.katneu)/max(y) - 1)) x.kat.ref <- c(x.kat.ref,
+               i * max(y) + 1)
+           for (i in x.kat.ref) x.katneu[i:(i + max(y) - 1),
+               ] <- t(t(x.katneu[i:(i + max(y) - 1), ]) - x.katneu[i,
+               ])
```

```
+             x.katneu <- x.katneu[-x.kat.ref, ]
+         }
+         pi.fit <- cbind(1, exp(tcrossprod(xdneu, be) + if (!missing(x.katneu)) matrix(x.katneu %*%
+             ga, nrow = nrow(xneu), byrow = T) else 0))/apply(exp(tcrossprod(xdneu,
+             be) + if (!missing(x.katneu)) matrix(x.katneu %*%
+             ga, nrow = nrow(xneu), byrow = T) else 0), 1, function(x) sum(x) +
+             1)
+         y.fit <- apply(pi.fit, 1, function(x) which(x == max(x)))
+         if (!missing(yneu))
+             mscl <- length(which(y.fit != yneu))
+     }
+     xdl <- cbind(1, x)
+     benu <- rbind(0, be)
+     pen <- if (max(y) == 2)
+         sum(abs(be[, -1])) + sum(abs(ga))
+     else sum(apply(be[, -1], 2, function(x) sqrt(sum(x^2)))) +
+         sum(abs(ga))
+     loglik <- -sum(diag(tcrossprod(tcrossprod(xdl, benu), y.mat)) -
+         Om %*% ga + apply(tcrossprod(xdl, be) + if (!missing(x.kat)) matrix(x.kat %*%
+         ga, nrow = nrow(x), byrow = T) else 0, 1, function(x) -log(1 +
+         sum(exp(x))))) + lambda * pen
+     if (!missing(xneu) & !missing(yneu)) {
+         yneu.mat <- matrix(0, nrow = length(yneu), ncol = max(y))
+         for (i in 1:length(yneu)) {
+             yneu.mat[i, yneu[i]] <- 1
+         }
+         if (!missing(x.katneu)) {
+             y.list <- list()
+             y.mat.kat <- yneu.mat[, -1]
+             for (i in 1:length(yneu)) y.list[[i]] <- t(y.mat.kat[i,
+                 ])
+             y.mat.kat <- bdiag(y.list)
+             Om <- y.mat.kat %*% x.katneu
+         }
+         xdl <- cbind(1, xneu)
+         benu <- rbind(0, be)
+         loglik.pr <- -sum(diag(xdl %*% t(benu) %*% t(yneu.mat)) +
+             (apply(xdl %*% t(be) + if (!missing(x.katneu)) matrix(x.katneu %*%
+                 ga, nrow = nrow(xneu), byrow = T) else 0, 1,
+                 function(x) -log(1 + sum(exp(x))))) - if (!missing(x.katneu)) sum(Om %*%
+             ga) else 0) + lambda * pen
+         brier <- sum((yneu.mat - pi.fit)^2)
+     }
+     names.b <- "Intercept"
+     for (i in 2:jl.l) {
+         if (length(jl[[i]]) == 1) {
+             names.b <- c(names.b, paste("koeff", i - 1))
+         }
+         else {
+             for (j in 1:length(jl[[i]])) names.b <- c(names.b,
+                 paste(paste("koeff", i - 1), j + 1, sep = "_"))
+         }
+     }
+     dimnames(be)[[2]] <- names.b
+     if (!missing(x.kat)) {
+         names.g <- "gamma 1"
+         for (i in 2:length(ga)) names.g <- c(names.g, paste("gamma",
+             i))
+         names(ga) <- names.g
+     }
+     erg <- list(if (max(y) == 2) as.vector(be) else be, loglik,
+         lambda)
+     ind <- 4
+     indn <- c("koeff", "loglik.pen", "lambda")
+     if (!missing(x.kat)) {
```

```
+         erg[[ind]] <- ga
+         ind <- ind + 1
+         erg[[ind]] <- sd.ga
+         ind <- ind + 1
+         indn <- c(indn, "ga", "sd.ga")
+     }
+     if (!missing(xneu)) {
+         erg[[ind]] <- pi.fit
+         ind <- ind + 1
+         erg[[ind]] <- y.fit
+         ind <- ind + 1
+         indn <- c(indn, "pi.fit", "y.fit")
+     }
+     if (!missing(yneu) & !missing(xneu)) {
+         erg[[ind]] <- mscl
+         ind <- ind + 1
+         erg[[ind]] <- loglik.pr
+         ind <- ind + 1
+         erg[[ind]] <- brier
+         ind <- ind + 1
+         indn <- c(indn, "missklass", "loglik.pr", "brier")
+     }
+     if (mult | mlog) {
+         erg[[ind]] <- fit.mult
+         ind <- ind + 1
+         indn <- c(indn, "fit.unpen")
+     }
+     names(erg) <- indn
+     return(erg)
+ }
```

Anhang B

Inhalt der dieser Arbeit beigefügten CD:

- diese Masterarbeit („MA_Uhlmann.pdf“)
- `R`-Code:
  - Funktionen zum Algorithmus („Algorithmus.r“):
    - Funktion, in welcher der Algorithmus umgesetzt ist („`plregr`“);
    - interne Funktionen von `plregr` („`umw`“ und „`likeli.j`“)
    - Funktion zur Erzeugung von Pfaden („`plregr.seq`“)
    - Funktion zum Plotten dieser Pfade (”`plregr.plot`“)
    - Funktion zur Kreuzvalidierung („`plregr.cv`“)
    - Funktion zur Durchführung der Simulationen („`plregr.sim`)
  - Funktion zur Generierung der Daten für die Simulationen („Generierung.r“)
  - Durchführung der Simulationen („Simulationen.r“)
  - Analyse der realen Daten („RDAnalyse.r“)
  - eine Version von `plregr`, in welcher die „alternative“ (Kapitel 4.5) Variablenselektion nicht verwendet wird; Anhand dieser Version wird also ein Algorithmus nach der in Kapitel 4.4 vorgestellten Methodik durchgeführt („Algorithmus_ohne_Alt.r“).
- ein Ordner mit den Ergebnissen aus den Simulationen („Ergebnisse“)
- der reale Datensatz, welcher in Kapitel 5.2 verwendet wird („BTW09small.dta“)
- Beschreibung der Variablen des realen Datensatzes („VariablenDef BTW09small.docx“)
- Ergebnisse zum Datensatz („RD_Ergebnisse.RData“)
- Eine Datei, welche eine Übersicht über den Inhalt der CD gibt („Inhalt.pdf“)

Erklärung zur Erstellung dieser Arbeit

Ich, Lorenz Uhlmann (Matrikelnummer: 4071247), versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.