Masterthesis

Machine Learning auf anonymisierten Daten

- Ein Vergleich verschiedener ML-Ansätze und Anonymisierungsverfahren

Institut für Statistik
Ludwig-Maximilians-Universität München

Paula Alflen

München, 09 November 2022

zur Erlangung des Grades Master of Science in Statistik
betreut von Prof. Dr. Thomas Augustin

"Ein Geist, der sich für eine neue Idee öffnet, wird niemals zu seiner ursprünglichen Größe zurückkehren." (Albert Einstein)

Danksagung

Ich danke ganz herzlich Professor Dr. Thomas Augustin für die Gelegenheit, einen Beitrag auf diesem interessanten Forschungsgebiet leisten zu dürfen. Darüber hinaus bin ich ihm auch dankbar für die großartige Betreuung während dieser Thesis (in kürzerer Zeit und sogar auch mal an einem Samstag Abend - unbeschreiblich!), für das Teilen seiner wertvollen Weisheit mit mir sowie für seine Ermutigung, wenn meine Energie am Ende einmal den saturierten Zustand erreicht hat. Ebenso danke ich ihm für die Unterstützung während meines gesamten Masters. Als er mein erstes wissenschaftliches Essay anforderte, fühlte ich mich vor eine große Herausforderung gestellt. So habe ich eine Weile gebraucht, um den ersten Satz auf Deutsch zu schreiben, den ich nach Kurzem doch wieder löschen musste, da er nicht gut genug war. Nach diesem ersten kamen die nächsten Essays, welche mir nicht nur im Rahmen der Statistik, sondern auch in der deutschen Sprache unheimlich viel gebracht haben. Auch hierfür mein unendliches Dankeschön.

Ferner danke ich auch meiner Familie und meinen Freunden(innen), unabhängig von den räumlichen Distanzen, die uns trennen, für die Unterstützung in jeglicher Form in dieser Zeit. Auch für das Verständnis für meine Abwesenheit während der gesamten Zeit, in der ich in meinem Zimmer saß und an dieser Thesis arbeitete. Es bedeutet mir unheimlich viel. Besonders bedanken möchte ich mich bei meinen Eltern, die ich solange mein Herz schlägt stolz machen und denen ich meine Leistungen widmen möchte.

Zudem danke ich auch herzlich Dana Hailer und Alexander Block, sehr netten Kollegen aus der Arbeit, sowie Jessica Drescher, die sich für das Korrekturlesen angeboten und ihre wertvolle Zeit hierfür investiert haben. Über ihre Bereitschaft und Herzlichkeit habe ich mich total gefreut und ich wünsche mir, dass das Universum ihnen zehn mal so viel Gutes zurückschickt.

Ebenfalls danke ich Gott, der mich nicht hat aufhören lassen, besonders in den schwierigen Zeiten, und der es mir somit ermöglicht hat, meinen Traum - meinen Master in Deutschland zu machen - in Erfüllung gehen zu lassen. Zudem auch dafür, dass er mir erlaubt hat, so großartigen Menschen zu begegnen.

Zusammenfassung

Der Schutz der Privatsphäre ist in der modernen Gesellschaft von heute zu einem immer wichtigeren Anliegen geworden. Die Statistical Disclosure Control Methoden kommen in diesem Kontext zum Einsatz, indem sie versuchen, einen Trade-off zwischen Offenlegungsrisiko und Nutzen der Daten bzw. Informationsverlust zu erreichen. Mit dieser Aufgabenstellung im Fokus wird sich diese Masterthesis mit der Anwendung verschiedener Machine Learning Ansätze auf mit diversen Anonymisierungsverfahren perturbierten Daten auf Mikroebene beschäftigen. Es wird untersucht, wie empfindlich sich Machine Learning Algorithmen auf anonymisierte Daten auswirken, wie deren Ergebnisse auf unterschiedlichen Leveln von Messfehlern und Fehlklassifikationen sind, welche Effekte das (absichtliche oder unabsichtliche) Perturbieren der Daten mit sich bringt, wie aus diesen perturbierten Daten dennoch Informationen gewonnen werden können, sowie wie sie adjustiert bzw. korrigiert werden können, um valide Analysen bzw. Inferenz zu betreiben.

Als Ergebnis für die metrische perturbierte Kovariable sind folgende Punkte resümiert: 1) Ist eine Kovariable unwichtig (unabhängig von der Korrelationsstruktur) oder normalverteilt oder werden Neuronale Netze oder Microaggregation angewendet, sind keine erheblichen Steigerungen des Root Mean Square Errors zu erwarten; 2) Letztgenannter erhöht sich dagegen deutlich mit der Steigerung der Variable Importance bei einer (korrelierten oder unkorrelierten) nicht normalverteilten Kovariable.

Ferner werden ein Korrekturansatz mittels Simulation-Extrapolation Methode hinsichtlich β sowie einer mittels Regressionskalibrierung für die individuellen Prädiktionen y_i bei metrischer mit additivem Fehler perturbierter Kovariable in einem linearen Regressionsmodell dargelegt. Korrekturansätze hinsichtlich β bei Adjustierung der Likelihood Funktion im Rahmen einer binären perturbierten Zielvariable sowie Kovariable werden ebenso präsentiert. In diesem Zusammenhang werden zwei weitere Korrekturmethoden hinsichtlich Adjustierung individueller Prädiktionen y_i entwickelt und dargestellt; eine davon war erfolgreich.

Abkürzungsverzeichnis

AIC	Akaike Information Criterion
bzw.	beziehungsweise
CSS	Complex Survey Samples
d.h.	das heißt
DR	Disclosure Risk
EM	Expectation-Maximization
EUSILC	European Union Statistics on Income and Living Conditions
FN	False Negative
FP	False Positive
GBM	Gradient Boosting Machine
IL	Information Loss
KNN	K-Nächste-Nachbarn
LM	Lineares Modell
MAD	Median Absolute Deviation
MAE	Mean Absolute Error
MCD	Minimum Covariance Determinant
MCEM	Monte Carlo Expectation-Maximization
MCSIMEX	MisClassification Simulation-Extrapolation
MDAV	Maximum Distance to Average
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NB	Nebenbemerkung
NN	Neuronale Netze
NSI	National Statistical Institute
PCA	Principal Component Analysis
PPV	Positive Predictive Values
PRAM	Post-Randomization Method
RC	Regression Calibration
RF	Random Forest
RMD	Robust Mahalanobis Distance
RMSE	Root Mean Squared Error
ROC	Receiver Operating Characteristic
ROMM	Random Orthogonal Matrix Masking
RR	Randomised Response
S.	Seite
SDC	Statistical Disclosure Control
SIMEX	Simulation-Extrapolation
SSE	Sum of Squares Error
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
u.a.	unter anderem
usw.	und so weiter
vgl.	vergleich
vs.	versus
XAI	Explainable Artifical Intelligence
z.B.	zum Beispiel

Inhaltsverzeichnis

Danksagung **i**

Zusammenfassung **ii**

Abkürzungsverzeichnis **iii**

1 Einleitung **1**

2 SDC und Offenlegungsrisiko **5**

2.1 Motivation / Bedeutung der Verringerung 5

2.2 Informationsverlust vs. Offenlegungsrisiko 6

2.3 Arten von Variablen 7

2.4 K-Anonymität 9

2.5 L-Diversität 9

2.6 Metriken für die Schätzung des Offenlegungsrisikos 10

2.7 Metriken für die Schätzung des Informationsverlusts 12

3 Anonymisierungsverfahren **13**

3.1 Methoden für kategoriale Variablen 14

3.1.1 Recoding 14

3.1.2 Local Suppression 15

3.1.3 Post-Randomization Methode (PRAM) 16

3.2 Methoden für metrische Variablen 20

3.2.1 Microaggregation 20

3.2.1.1 *Individual Ranking* Methode 22

3.2.1.2 *MDAV* Methode 22

3.2.1.3 *RMD* Methode 23

3.2.1.4 *Gower Distance* Methode 25

3.2.1.5 *PCA* Methode 25

3.2.1.6 *Simple* Methode 28

3.2.2 Noise Addition 29

3.2.2.1 *Additive* Methode 29

3.2.2.2 *Correlated* Methode 32

3.2.2.3 *Correlated2* Methode 34

3.2.2.4 *Outdect* Methode 36

4 Grundlagen in Machine Learning **37**
4.1 Fundament . . . 37
4.2 Algorithmen . . . 40
4.2.1 Lineares Modell (LM) . . . 40
4.2.2 K-Nächste-Nachbarn (KNN) . . . 41
4.2.3 Random Forest (RF) . . . 41
4.2.4 Gradient Boosting Machine (GBM) . . . 42
4.2.5 Neuronale Netze (NN) . . . 43

5 Anwendung von ML-Algorithmen und Anonymisierungsverfahren **46**
5.1 Metrische Zielvariable . . . 46
5.1.1 Daten . . . 46
5.1.2 Anwendung von Anonymisierungsverfahren . . . 49
5.1.3 Anwendung von ML-Algorithmen . . . 51
5.1.4 Performance Metriken . . . 52
5.2 Binäre Zielvariable . . . 53
5.2.1 Logistische Regression und Newton-Raphson-Algorithmus . . . 53
5.2.2 Perturbation einer binären Zielvariable . . . 54
5.2.3 Perturbation einer binären Kovariable . . . 55
5.2.4 EM-Algorithmus . . . 56

6 Ergebnisse **58**
6.1 Metrische Zielvariable . . . 58
6.1.1 Fall betrachtete Ergebnisse hinsichtlich RMSE . . . 58
6.1.2 Allgemeine Schlussfolgerungen der Ergebnisse hinsichtlich RMSE . 59
6.1.3 Ergebnisse der Variable Importance . . . 64
6.1.4 Ergebnisse Offenlegungsrisiko vs. Informationsverlust . . . 70
6.1.5 (β)-Korrekturansatz bei metrischer X^* mittels SIMEX Methode . . 72
6.1.6 Korrekturansatz für individuelle Prädiktionen y_i bei metrischer X^* 77
6.2 Binäre Zielvariable . . . 79
6.2.1 (β)-Korrekturansatz für Y^* mittels Newton-Raphson-Algorithmus . 79
6.2.2 (β)-Korrekturansatz für X^* mittels EM-Algorithmus . . . 82
6.2.3 Korrekturansatz für individuelle Prädiktionen y_i bei binärer X^* . . 84
6.2.4 Korrekturansatz für individuelle Prädiktionen y_i bei binärer Y^* . . 86

7 Konklusion und Ausblick **89**

Literatur **92**

Elektronischer Anhang **97**

Abbildungsverzeichnis **98**

Tabellenverzeichnis **100**

Declaration of authorship **102**

1 Einleitung

Den Geist einladen, sich für eine neue Idee zu öffnen

Domingo-Ferrer et al. (2012) zufolge verfügen statistische Ämter heutzutage über sehr große Datenbanken zur Erstellung ihrer Statistiken, was einen großen Vorteil mit sich bringt. Wie Domingo-Ferrer et al. (2012, S. xii) betonen, können diese großen Datenbanken ein "second life" bekommen, indem sie als Grundlage für weitere statistische Forschungsprojekte dienen, was sowohl zur Einsparung von Zeit als auch Geld führt, denn "collaboration is much more efficient"(idem). Bevor die Daten jedoch dafür zur Verfügung gestellt werden können, muss die Vertraulichkeit von sensiblen Daten gewährleistet sein (idem).

In diesem Zusammenhang kommen die Statistical Disclosure Control (SDC) Techniken zur Anwendung. Sie zielen darauf ab, die Daten so zu behandeln und zu modifizieren, dass ein akzeptables Offenlegungsrisiko erreicht wird. Somit können die Daten veröffentlicht bzw. freigegeben werden, ohne dass die zugrunde liegenden vertraulichen Informationen erkannt bzw. mit jemandem assoziiert werden können, während gleichzeitig der Informationsverlust durch die Anonymisierung minimal sowie der Nutzen der Daten maximal wird (vgl. Benschop and Welch, 2019; Domingo-Ferrer et al., 2002, S. 1). "The level of acceptability of disclosure risk and the need for anonymization are usually at the discretion of the data producer and guided by legislation"(Benschop and Welch, 2019).

Darauf basierend heben Caiola and Reiter (2010) einen essenziellen Punkt hervor, den ich an dieser Stelle gern zitieren möchte:

> *"These methods can be applied with varying intensities. Generally, increasing the amount of alteration decreases the risks of disclosures; but, it also decreases the accuracy of inferences obtained from the released data, since these methods distort relationships among the variables."* (Caiola and Reiter, 2010, S. 28)

Ferner bildet Yi (2017, S. 44) eine Analogie zwischen Anonymisierung und Messfehler wie folgt: Anonymisierung von Daten kann auch als eine Art Messfehler bzw. Fehlklassifikation betrachtet werden, indem unpräzise Messungen absichtlich aus ethischen Gründen generiert werden. Ferner argumentiert Wallace (2020, S. 16), dass "truly perfect measurements may be impossible to take". Um die Situation näher zu beleuchten, kann

Folgendes angenommen werden. Es liegen Daten in Form von Y (metrische oder kategoriale Zielvariable) und X (metrische oder kategoriale Kovariablen) vor. Von Interesse ist der Zusammenhang zwischen X und Y, der Effekt von X auf Y oder Y mittels X vorherzusagen. Jedoch können die exakten Werte von X bzw. Y nicht beobachtet werden (was zu Messfehlern führt) oder dürfen aufgrund des Datenschutzes nicht in ihrem ursprünglichen Zustand verwendet werden (was zur Anonymisierung führt). Die wahren Variablen X bzw. Y werden dann durch ihre Surrogate, die mit Messfehlern bzw. Perturbationen kontaminierten X^* bzw. Y^*, ersetzt. "For years, measurement error has been ignored because of the claim that its impact is "not that bad"" (Wallace, 2020, S. 16). Daraus folgen allerdings diverse "Measurement Error Effects" (Yi, 2017, S. 45) und "to assume our measurements are perfect (or sufficiently "near perfect") when in reality they are not can have grave statistical consequences" (Wallace, 2020, S. 14), welche von vielen Faktoren abhängen, "including the form of the response and measurement error models, the variability of the variables, and their association structures" (Yi, 2017, S. 48).

Unabhängig von den Gründen und Quellen der Messfehler bzw. Perturbationen, treten einige Fragen auf: Wie lassen sich aus diesen kontaminierten Daten dennoch Informationen gewinnen bzw. Zusammenhänge erkennen? Wie kann valide statistische Inferenz daraus folgen? Welche Effekte verursachen die Anonymisierungsverfahren bzw. Messfehler auf die Variablen? (Wie) können die kontaminierten Daten adjustiert bzw. korrigiert werden, um die Perturbation zu berücksichtigen?

Ziel der Masterthesis

Da Machine Learning (ML) auch die Statistik umfasst, indem Zusammenhänge zwischen Variablen und allgemeinen Strukturen erkannt werden können, ist das Ziel der vorliegenden Masterthesis wie folgt: Es wird untersucht, wie sensibel ML-Algorithmen auf anonymisierte Daten reagieren, wie empfindlich deren Ergebnisse auf unterschiedlichen Leveln von Messfehlern und Fehlklassifikationen sind und welche möglichen Faktoren deren Performance beeinflussen. Diese Masterthesis befasst sich ausschließlich mit Daten auf Mikroebene, d.h. eine Datenmatrix, in der jede Zeile einer Beobachtung und jede Spalte einer Variable entsprechen. Zunächst werden die Grundlagen der Anonymisierungsverfahren und ML-Ansätze vorgestellt. Anschließend werden beide in praktischen Fällen mit metrischer sowie binärer Zielvariable angewendet und abschließend ihre Performance bzw. Verhalten evaluiert. Die Arbeit schließt mit der Darstellung möglicher Korrekturverfahren für die untersuchten, durch die Perturbation verursachten, Sensibilität bzw. Messfehler ab.

Praktische Anwendung

Die praktische Anwendung kann wie folgt in die zwei Komponenten metrische und binäre Zielvariable aufgeteilt werden:

1. Metrische Zielvariable (unterteilt in Anwendung der Anonymisierungsverfahren, der ML-Ansätze, allgemeine Simulationen und Korrekturansatz):

   (a) es werden zunächst verschiedene Anonymisierungsverfahren auf einige Kovariablen in einem ursprünglichen Datensatz, nämlich dem EUSILC Datensatz, angewendet (*NB: Die Outcome Variable wird nicht kontaminiert*). Dazu kommen folgende Anonymisierungsverfahren zur Anwendung: i) Recoding; ii) Microaggregation mit der Maximum Distance to Average ($MDAV$) Methode sowie Variation des Aggregationslevels, nämlich mit 3, 10 und 15; iii) Noise Addition mit Variation sowohl der Methoden, nämlich *additive-* sowie *correlated*2 Noise, als auch der Menge an Noise, nämlich 50, 100, 150, 200 und 250 (bei additiver Noise Methode) sowie bei *correlated*2 des α-Hyperparameters (nämlich 0.1, 0.25, 0.5, 0.75 und 1). Die zu perturbierenden Variablen wurden so gewählt, dass verschiedene Korrelationsstrukturen aufkommen und deren Verhalten untersucht werden kann. Local Suppression wird wegen des zu hohen Informationsverlusts durch das Generieren fehlender Werte nicht verwendet.

   (b) folgend werden einige ML-Algorithmen auf den eben genannten Datensatz mit metrischer Zielvariable angewendet, und deren Performance anschließend mit einigen geeigneten Metriken evaluiert. Dazu werden die Algorithmen Lineares Modell, K-Nächste-Nachbarn, Random Forest, Gradient Boosting Machine und Neuronale Netze verwendet. Diese werden zunächst auf die originalen nichtanonymisierten Daten angewendet und dann auf die mit den unterschiedlichen Anonymisierungstools perturbierten Daten. Deren Root Mean Squared Error (RMSE), R-squared, Mean Absolute Error (MAE) und Bias werden berechnet und anschließend mit Erstgenanntem werden sie evaluiert. Die Variable Importance bei Random Forest und Gradient Boosting Machine ist ebenso von Bedeutung.

   (c) Um die Fall betrachteten Ergebnisse des EUSILC Datensatzes verallgemeinern zu können, wurden zwei Simulationen durchgeführt, deren Ergebnisse am Schluss beschrieben werden. Eine dritte Simulation erfolgte, um das Offenlegungsrisiko vs. Informationsverlust zu untersuchen. Die Implementierung erfolgte mithilfe der statistischen Software `R` (R Core Team, 2022) sowie der später erwähnten Packages.

(d) Zum Schluss dieses Teils wurden zum einen ein Korrekturverfahren mittels Simulation-Extrapolation (SIMEX) Methode hinsichtlich β und zum anderen ein mittels Regressionskalibrierung für die individuellen Prädiktionen y_i bei metrischer mit additivem Fehler perturbierter Kovariable in einem linearen Regressionsmodell dargelegt. Der Korrekturansatz wurde auch in `R` durchgeführt und evaluiert.

2. Binäre Zielvariable:
   Die binäre Zielvariable wurde überwiegend auf mathematische Weise betrachtet und dargelegt, und zudem eine Logistische Regression herangezogen. Dazu wurden zwei Fälle betrachtet: Eine binäre durch PRAM perturbierte zum einen Zielvariable und zum anderen Kovariable. In beiden Fällen wurden die perturbierten Likelihood Funktionen dargestellt. Als nächster Schritt erfolgte deren Korrektur, indem sie geeignet angepasst wurden, um die mittels PRAM verursachte Perturbation zu erfassen. Bei der perturbierten Zielvariable kam diesbezüglich der Newton-Raphson-Algorithmus zum Einsatz, während bei der perturbierten Kovariable der EM- bzw. Monte Carlo EM-Algorithmus zur Anwendung kam. Die Korrekturen geschahen dabei im Hinblick auf die βs, welche in gewisser Weise den gesamten Zusammenhang gestalten bzw. steuern. Danach wurden zwei Korrekturmethoden entwickelt, welche die individuellen Prädiktionen (d.h. y_i) versuchen zu korrigieren, die zum einen eine binäre perturbierte Outcome Variable und zum anderen eine binäre kontaminierte Kovariable betrachten. Die Idee dahinter ist, die Genauigkeit zu optimieren, indem die Beobachtungen mit den meisten Effekten auf der Genauigkeit korrigiert werden.

Gliederung

Die vorliegende Masterthesis ist folgendermaßen strukturiert: Zunächst wird das Konzept der Statistical Disclosure Control (SDC) vorgestellt. Anschließend werden das Offenlegungsrisiko sowie damit verbundene Begriffe wie Informationsverlust, K-Anonymität und L-Diversität näher beschrieben. Danach werden einige Metriken der Schätzung des Offenlegungsrisikos bzw. des Informationsverlusts dargestellt (siehe Kapitel 2). Dann erfolgt in Kapitel 3 eine ausführliche Erläuterung verschiedener entwickelter Anonymisierungsverfahren, die auch für das weitere Vorgehen relevant sind. Anschließend befasst sich Kapitel 4 mit den Grundlagen in Machine Learning sowie der Beschreibung der verwendeten ML-Algorithmen. Als Nächstes wird die Anwendung von ML-Ansätzen und Anonymisierungsverfahren für eine metrische sowie binäre Zielvariable beschrieben (siehe Kapitel 5). Auf dieser Grundlage können in Kapitel 6 die Ergebnisse präsentiert und evaluiert werden. Darüber hinaus finden hier auch die Korrekturansätze Raum. Das Kapitel 7 schließt die Arbeit mit einer Zusammenfassung der Ergebnisse und einem Ausblick ab.

2 SDC und Offenlegungsrisiko

Der Fokus dieses Kapitels liegt auf der Statistical Disclosure Control, auf dem Offenlegungsrisiko, auf der Bedeutung seiner Verringerung (2.1), dem Trade-off zwischen Risiko und Informationsverlust (2.2), auf den Arten von Variablen (2.3), auf K-Anonymität (2.4) sowie L-Diversität (2.5). Diese Begriffe sind für die weiteren Kapitel relevant. Des Weiteren werden Metriken für die Schätzung des Offenlegungsrisikos (2.6) sowie Informationsverlusts präsentiert (2.7), welche in 6.1.4 zur Anwendung kommen.

2.1 Motivation / Bedeutung der Verringerung

Eine Offenlegung liegt vor, wenn eine Person oder eine Organisation eine andere Person (Offenlegung der Identität) bzw. ein Attribut von ihr (Offenlegung von Attributen) erkennt, erfährt, re-identifiziert, assoziiert bzw. zuordnen kann, was vor den freigegebenen vertraulichen Daten noch nicht bekannt war (siehe z.B. Hundepool et al., 2006, S. 7; Domingo-Ferrer et al., 2012, S. 2-3).

Die Bedeutung des Schutzes personenbezogener Daten wird in Domingo-Ferrer et al. (2012, S. xi-xii) hervorgehoben. Beginnend mit den rechtlichen Rahmenbedingungen, welche regeln, was bezüglich der Veröffentlichung privater Informationen erlaubt und was nicht erlaubt ist, muss des Weiteren eine gute Beziehung zwischen Datenverarbeiter und Betroffenen gepflegt werden, denn die Letztgenannten sind die Quelle, auf deren Statistiken aufgebaut werden. "The respondents must be able to trust that their private and often sensitive information is safe in the hands of statistical offices"(Domingo-Ferrer et al., 2012, S. xii). Haben die Betroffenen das Gefühl, ihre Privatsphäre sei nicht geschützt, sind sie möglicherweise nicht bereit, an künftigen Umfragen teilzunehmen (Benschop and Welch, 2019).

Dies berücksichtigend liegt das Ziel der Statistical Disclosure Control (SDC) darin, statistische Daten so zu schützen, dass sie der Öffentlichkeit zur Verfügung gestellt werden können, ohne vertrauliche Informationen preiszugeben, die mit bestimmten Personen oder Organisationen verlinkt werden können (Domingo-Ferrer et al., 2012, S. 1). "SDC methods minimise the risk of disclosure to an acceptable level while releasing as much information as possible"(Domingo-Ferrer et al., 2012, S. 3). Benschop and Welch (2019) zufolge werden

somit die Datenqualität sowie die Responsequoten zukünftiger Erhebungen garantiert. Diverse Methoden und Verfahren wurden zu diesem Zweck entwickelt; einige davon werden in Kapitel 3 näher beschrieben.

Ferner ist der Nutzen der Daten zu berücksichtigen. Wird auf die Daten übermäßiger bzw. unangemessener SDC-Schutz bzw. SDC-Anonymisierung appliziert, sind sie nicht mehr (ausreichend) anwendbar und die ganze (zeitliche bzw. finanzielle) Investition war umsonst (Benschop and Welch, 2019). Damit befasst sich das nächste Unterkapitel 2.2 ausführlicher.

2.2 Informationsverlust vs. Offenlegungsrisiko

Durch SDC-Methoden wird das Offenlegungsrisiko nicht komplett eliminiert, kann aber auf ein akzeptables Niveau reduziert werden (siehe z.B. Domingo-Ferrer et al., 2012, S. 8-9; Benschop and Welch, 2019). Dabei führt die Anwendung von SDC-Methoden auf die ursprünglichen Daten zu einem Informationsverlust, was wiederum die Qualität der Daten reduziert (vgl. Templ, 2017, S. 42).

Generell gilt: "the lower the disclosure risk, the higher the information loss and the lower the data utility"(siehe z.B. Benschop and Welch, 2019; Templ, 2017, S. 43-44). Auf Grund dessen besteht die größte Herausforderung der Anwendung von SDC-Techniken darin, den optimalen Trade-off zu erreichen, indem die Offenlegungsrisiken verringert werden, bei minimalem Informationsverlust und maximalem Nutzen der Daten (siehe z.B. Templ, 2017, S. 43; Hundepool et al., 2006, S. 8-9; Domingo-Ferrer et al., 2012, S. 4).

Templ (2017, S. 43) bringt ein Beispiel, indem dieser Trade-off illustriert wird, hier in Abbildung 1 dargestellt. Es wird angenommen, die originalen Daten haben 0% Informationsverlust sowie 100% Offenlegungsrisiko. Bei der Abbildung 1 startet die Kurve somit bei (1, 0). Dabei werden drei Anonymisierungsverfahren verwendet, welche auch in Kapitel 3 zu finden sind, nämlich Microaggregation (3.2.1.2), *additive* sowie *correlated* Noise Addition (Unterkapitel 3.2.2.1 und 3.2.2.2). Daraus lässt sich ablesen, je mehr Anonymisierung zur Anwendung kommt (d.h. von rechts ausgehend nach links), desto kleiner ist das Offenlegungsrisiko, gleichzeitig aber umso höher der Informationsverlust.

In den Unterkapiteln 2.4 und 2.5 werden einige Begriffe im Rahmen dieses Trade-offs kurz vorgestellt, nämlich K-Anonymität und L-Diversität. Weitere wie z.B. Differential Privacy werden in u.a. Domingo-Ferrer et al. (2012, S. 6-7) angesprochen. Davor wird noch in Unterkapitel 2.3 eine zusätzliche Erklärung der Arten der Variablen eingeführt, welche für das weitere Verständnis von Bedeutung sind.

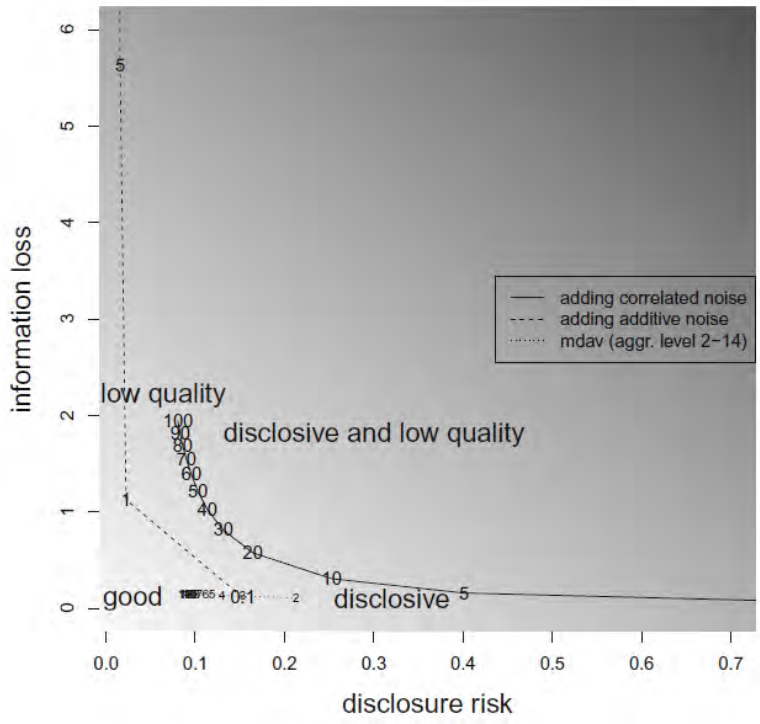


Abbildung 1: Trade-off zwischen Informationsverlust und Disclosure Risk (aus Templ (2017, S. 43) entnommen)

2.3 Arten von Variablen

Im Kontext von Statistical Disclosure Control (SDC) gibt es verschiedene Arten von Variablen, deren Unterscheidung wichtig zu verstehen ist, wie die Templ (2017) entnommene Abbildung 2 illustriert:

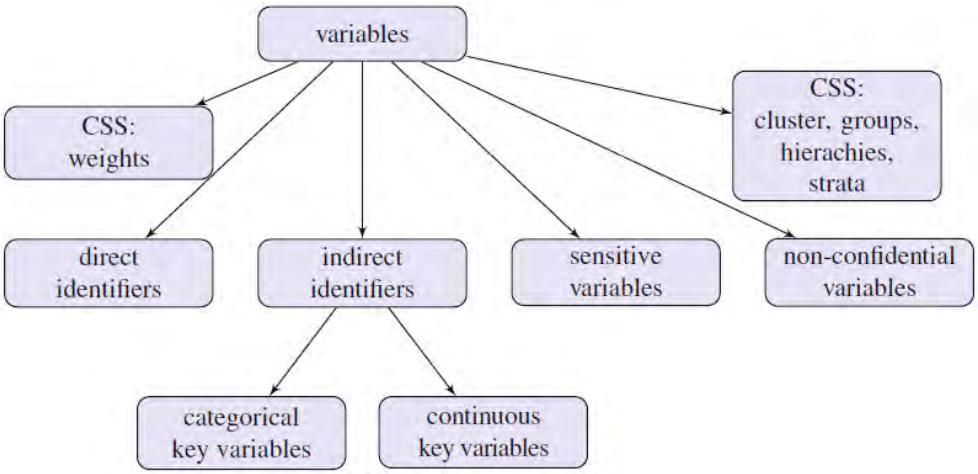


Abbildung 2: Arten von Variablen (aus Templ (2017, S. 36) entnommen)

- **Non-confidential variables:** Für diese Variablen wird angenommen, dass keine Information in externen Datensätzen zu finden ist (vgl. Templ, 2017, S. 35).

- **Direct identifiers variables:** Bei dieser Art wird von einzelnen Variablen ausgegangen, dass deren (einzelne) Werte zur eindeutigen Re-Identifikation der Personen bzw. Beobachtungen führt. Beispiele sind Sozialversicherungsnummern, Namen, Adressen und IDs von Personen und Unternehmen. Deshalb ist der erste Schritt bei SDC-Methoden, diese Variablen aus dem Datensatz zu entfernen (vgl. Templ, 2017, S. 36).

- **Indirect identifiers / key variables:** Anders als bei direct identifiers, ist hier zwar eine einzelne Variable hinsichtlich des Veröffentlichungsrisikos nicht problematisch, allerdings kann sie in Kombination mit anderen dazu führen, dass Personen erkannt werden. Indirect identifiers Variablen werden auch (kategoriale bzw. kontinuierliche) Key Variablen genannt. Zum Beispiel: Obwohl Alter, Geschlecht, Religion und Beruf jede für sich keine Zuordnung erlauben, können durch deren Kombination Personen re-identifiziert werden (idem).

- **Sensitive variables:** Für diese Variablen dürfen die Werte bei keinem Befragten im Datensatz re-identifiziert werden. Eine Variable kann sowohl sensitive als auch direct identifier bzw. Key Variable sein. Beispiele dafür sind Variablen, in denen Information zur kriminellen Historie, sexuellem Verhalten, medizinischen Daten oder Einkommen enthalten sind (vgl. Templ, 2017, S. 36-37).

- **Sampling Weights:** Eher bei Haushaltserhebungen auftretend, nutzen Sampling Weights (mindestens) einen Vektor mit individuellen Gewichten, d.h. die Chancen für jedes Individuum für die Teilnahme an der Befragung ausgewählt zu werden. Falls eine Stichprobe anstelle einer ganzen Population betrachtet wird, kommt eine weitere Unsicherheit hinzu, da normalerweise nicht bekannt ist, ob eine bestimmte Person an der Umfrage teilgenommen hat. Außer sie hat eindeutige Werte in der Population, denn Templ (2017) zufolge beträgt deren Offenlegungsrisiko somit dasselbe.

  > *"Sampling has an effect on the disclosure risk and are taken into account for estimation of the disclosure risk. [...] For instance, single-parent households will be given a greater chance of being selected in a survey with the aim of estimating low incomes, and weights can adjust for this."* (Templ, 2017, S. 37)

- **Hierarchies / Cluster variables:** Weisen die Daten Hierarchien bzw. Cluster auf, muss dies auch bei den SDC-Methoden berücksichtigt werden. Beispiel hierfür ist

die Austrian Structural Earning Statistics Erhebung, in der zunächst Unternehmen ausgewählt werden und dann deren Arbeitnehmer in der zweiten Stufe (vgl. Templ, 2017, S. 38).

2.4 K-Anonymität

Gibt es eindeutige Beobachtungen in einem Datensatz, müssen diese stärker geschützt werden, um das Re-Identifikationsrisiko zu reduzieren. Ein Ansatz in diesem Kontext ist die K-Anonymität. "An individual violates k-anonymity if the sample frequency count f_k for the key [...] is smaller than the specified threshold k"(Benschop and Welch, 2019). K-Anonymität ist ein Risikomaß basierend auf dem Prinzip, dass die Anzahl an Personen bzw. Einheiten, die das gleiche Muster in einem Datensatz teilen, nämlich die Kombination der Key Variablen (beschrieben in Unterkapitel 2.3), wenigstens gleich dem festzulegenden Hyperparameter k oder sogar höher ist (siehe z.B. Benschop and Welch, 2019; Templ, 2017, S. 58). K dient somit als ein Threshold, der garantiert, dass keine eindeutigen Beobachtungen zu finden sind, was zum Erreichen von K-Anonymität führt. Übliche festzulegende Werte für k sind 2, 3 und 5.

Der Begriff wird mit folgendem Beispiel 1 verdeutlicht:

	Land	Geschlecht	Bildungsgrad
1	Deutschland	weiblich	Master
2	Schweiz	männlich	Master
3	Deutschland	weiblich	Bachelor
4	Deutschland	weiblich	Master
5	Schweiz	männlich	Master

Tabelle 1: Beispiel für die Verletzung von K-Anonymität

Beobachtung 3 in der obigen Tabelle ist eindeutig mit der Kombination aus Land = "Deutschland", Geschlecht = "weiblich" und Bildungsgrad = "Bachelor" in diesem kleinen Datensatz, was somit sogar 2-Anonymität verletzt und ein hohes Offenlegungsrisiko aufweist. Um Letztgenanntes zu reduzieren, können einige der Anonymisierungsverfahren angewendet werden, welche näher in Kapitel 3 beschrieben sind. Hier, um eine einfache Lösung vorzuschlagen, könnte z.B. die Ausprägung "Bachelor" bei dieser Beobachtung nach "Master" umkodiert werden.

2.5 L-Diversität

Während K-Anonymität auf den Key Variablen basiert, beruht L-Diversität auf sensitiven Variablen (in Unterkapitel 2.3 beschrieben). Auch wenn die Beobachtungen K-Anonymität

nicht verletzen, können sensitive Informationen über bestimmte Individuen extrahiert werden. Deshalb mag K-Anonymität alleine nicht einschränkend genug sein (Domingo-Ferrer et al., 2012, S. 6; Templ, 2017, S. 58; Benschop and Welch, 2019).

Folgendes Beispiel 2, aus Templ (2017, S. 58) entnommen, erläutert die Idee näher:

	Key Variables			Sensitive Variable	
	Gender	Age group	f_k	Medical condition	L-Diversity
1	Male	30s	3	Cancer	2
2	Male	30s	3	Heart disease	2
3	Male	30s	3	Heart disease	2
4	Female	20s	3	Cancer	1
5	Female	20s	3	Cancer	1
6	Female	20s	3	Cancer	1

Tabelle 2: Beispiel für die Verletzung von L-Diversität (aus Templ (2017, S. 58) entnommen)

Mit obiger Tabelle wird 3-Anonymität mit der Kombination der Key Variablen Gender und Age group erreicht, auch explizit in der Häufigkeitsspalte f_k zu sehen. Templ (2017, S. 58) bringt folgende Situation auf: Angenommen, ein Angreifer erhält diesen kleinen Datensatz und weiß zusätzlich, dass seine 20-jährige Nachbarin vor Kurzem in diesem Krankenhaus war. Da alle 20-jährigen Frauen Krebs haben, findet er ihren gesundheitlichen Status heraus, nämlich Krebs.

Um dieses Manko zu beheben, kommt L-Diversität zur Anwendung, welche als eine "stronger notion of privacy" eingeführt wurde (idem). Letztgenanntes stellt sicher, dass die sensitiven Variablen wenigstens l verschiedene Werte in der Kombination der Key Variablen haben (Templ, 2017, S. 59; Benschop and Welch, 2019). Um noch bei dem Beispiel 2 zu bleiben, wird bei den ersten drei Beobachtungen 2-Diversität erreicht, da deren Werte für die sensible Variable Medical condition aus zwei verschiedenen Ausprägungen bestehen, nämlich Cancer und Heart disease.

2.6 Metriken für die Schätzung des Offenlegungsrisikos

Bisher wurden die Begriffe des Offenlegungsrisikos und des Trade-offs zwischen der Anonymisierung und dem Informationsverlust dargestellt. Wie erfolgen deren Schätzungen? Die Messung bzw. Schätzung des Risikos ist eine wesentliche Aufgabe, um entscheiden zu können, ob die zu veröffentlichenden Daten ausreichend geschützt sind (vgl. Templ, 2017, S. 49-50). In diesem Unterkapitel werden nun einige Metriken diesbezüglich eingeführt.

Das Offenlegungsrisiko beruht auf Annahmen zu Offenlegungsszenarien (Templ, 2017, S. 49). Die Messung besteht darin festzustellen, ob eine Person bzw. eine Information von

ihr mit den perturbierten Daten verlinkt werden kann. Die Metriken zur Schätzung des Offenlegungsrisikos unterscheiden sich Templ (2017) zufolge, zwischen den kategorialen und den metrischen Variablen. Templ (2017, S. 50-90) stellt diverse Metriken ausführlicher dar, u.a. K-Anonymität und L-Diversität (bereits in Unterkapitel 2.4 und 2.5 diskutiert), Häufigkeitsauszählung, Special Uniques Detection Algorithmus, Individual Risk Ansatz, Offenlegungsrisiko für hierarchische Daten und globale Risiken. Das Risiko bei metrischen Variablen kann mittels Record Linkage- oder Intervalloffenlegungsansatz gemessen werden. Bei Erstgenanntem gibt es zwei Arten: Distance-based Record Linkage und Probabilistic Record Linkage. Diese werden im Folgenden kurz beschrieben:

- **Distance-based Record Linkage:** Dies ist ein Record Linkage Ansatz basierend auf Distanzen. Für jede Beobachtung im perturbierten Datensatz wird ihre Distanz zu jeder Beobachtung im originalen berechnet und die am nächsten (d.h. mit der kleinsten euklidischen Distanz) wird weiter berücksichtigt. Wenn die perturbierte und deren nächste ursprüngliche Beobachtung der gleichen Person entspricht, wird es als "linked" markiert. Das Offenlegungsrisiko wird somit als Prozentsatz der "linked" Beoachtungen im perturbierten Datensatz definiert (Templ, 2017, S. 91; Dandekar et al., 2002, S. 157; Domingo-Ferrer et al., 2001, S. 9; Mateo-Sanz et al., 2004, S. 204).

- **Probabilistic Record Linkage:** Der Matching-Algorithmus basiert auf einem Gewicht (d.h. die Likelihood) für jedes Paar, dass die zwei Beobachtungen (aus dem perturbierten und aus dem ursprünglichen Datensatz) sich auf denselben Befragten beziehen. Paare mit einem Gewicht höher als ein bestimmter Threshold werden ebenso als "linked" markiert und der Prozentsatz der "linked" Beobachtungen wird analog zu Distance-based Record Linkage als Offenlegungsrisiko definiert (vgl. Templ, 2017, S. 91; Domingo-Ferrer et al., 2001, S. 9-10).

- **Intervalloffenlegung:** Basierend auf dem Wert einer perturbierten Variable wird untersucht, ob der entsprechende originale Wert in ein Intervall fällt, welches um den perturbierten Wert zentriert ist. Die Breite des Intervalls wird entweder auf der Rangordnung der Variable oder auf ihrer Standardabweichung bestimmt (Mateo-Sanz et al., 2004, S. 204-205; Domingo-Ferrer et al., 2001, S. 10; Templ, 2017, S. 91-92). Das Offenlegungsrisiko ist der Anteil der originalen Werte, die in das Intervall fallen, basierend auf einer Worst-Case-Szenario-Annahme (Templ, 2017, S. 92). Diese Variante kommt in Unterkapitel 6.1.4 zum Einsatz, um das Offenlegungsrisiko darzustellen.

2.7 Metriken für die Schätzung des Informationsverlusts

Neben den vorherigen Metriken hinsichtlich des Offenlegungsrisikos muss auch der Nutzen der Daten bzw. der Informationsverlust berücksichtigt werden. Metriken für die Schätzung des Informationsverlusts bewerten inwiefern die Anwendung von Anonymisierungsverfahren die Daten unbrauchbar machen bzw. deren Qualität beeinträchtigen. Yancey et al. (2002, S. 141) reden auch von einem "penalty Score" in diesem Zusammenhang.

Hinsichtlich kategorialer Variablen werden drei Arten von Metriken in Betracht gezogen: Direkter Vergleich von kategorialen Werten, Vergleich von Kontingenztafeln und entropiebasierte Metriken, welche ausführlicher in Domingo-Ferrer et al. (2001, S. 7) beschrieben sind. Templ (2017, S. 133-139) leistet ebenfalls einen Beitrag dazu.

Auch für metrische Variablen sind diverse Metriken bereits definiert worden, wie z.B. IL1, IL1s, IL2, IL3, IL4 und IL5, welche in Yancey et al. (2002, S. 141-142) näher beschrieben sind. Im Package `sdcMicro` von Templ et al. (2015) ist eine weitere Metrik implementiert worden, nämlich *eigen*, welche die relativen absoluten Differenzen zwischen den Eigenwerten der Kovarianzen der originalen sowie der perturbierten Variablen schätzt (Templ, 2017, S. 140).

Im Folgenden wird die IL1s dargestellt, die auch in Unterkapitel 6.1.4 für die Schätzung des Informationsverlusts zur Anwendung kommt. Die von Yancey et al. (2002, S. 141-142) vorgeschlagene "uniform and intrinsic scaling method" ist wie folgt definiert:

$$IL1s = \frac{1}{np} \sum_{j=1}^{p} \sum_{i=1}^{n} \frac{|x_{ij} - x_{ij}^*|}{\sqrt{2}\sigma_j} \tag{1}$$

wobei x_{ij} die j-te Variable in der i-ten Beobachtung im originalen Datensatz, x_{ij}^* die kontaminierte Version davon, σ_j die Standardabweichung der j-ten originalen Variable, n die Anzahl der im Datensatz enthaltenen Beobachtungen und p die Anzahl der Variablen sind. Templ (2017, S. 141) zufolge kann diese Metrik als skalierte Distanzen zwischen den originalen und den perturbierten Daten verstanden werden.

3 Anonymisierungsverfahren

In diesem Kapitel wird ein grundlegendes Verständnis der Anonymisierungsverfahren vermittelt, welche gemeinsam mit den Grundlagen aus Kapitel 2 als Fundament für die nachfolgenden Kapitel der Arbeit dienen soll.

Die Statistical Disclosure Control (SDC) Techniken können in drei Kategorien unterteilt werden (vgl. Templ, 2017, S. 99; Hundepool et al., 2006, S. 58-59):

- Non-perturbative Techniken, welche die Details supprimieren bzw. reduzieren, ohne die ursprünglichen Daten zu perturbieren bzw. zu verändern, wie z.B. Recoding und Local Suppression.
- Perturbative Techniken, welche die Daten de facto perturbieren bzw. verzerren, wie z.B. PRAM, Microaggregation und Noise Addition.
- Techniken für die Generierung eines synthetischen Datensatzes, welcher bedingte statistische Verhältnisse des originalen Datensatzes bewahrt.

Methoden zur Perturbation der Daten für kategoriale Variablen unterscheiden sich von denen für metrische Variablen. Eine kleine Anzahl an Beobachtungen in einer Kategorie kann ein Problem darstellen, da das Offenlegungsrisiko für kategoriale Variablen von deren Häufigkeiten abhängt (Templ, 2017, S. 99). In solchen Fällen wird häufig das Recodingsverfahren und, falls das Risiko danach noch zu hoch ist, Local Suppression angewendet (idem). Eine weitere Alternative bei großer Anzahl an Key Variablen ist die Post-Randomization Methode (PRAM). Was die Perturbation der Daten für metrische Variablen anbelangt, können Microaggregation bzw. Noise Addition verwendet werden.

Zunächst werden im Abschnitt 3.1 die Anonymisierungsverfahren für kategoriale Variablen genauer beschrieben. Anschließend beschäftigt sich der Abschnitt 3.2 mit der Beschreibung der Anonymisierungsmethoden für metrische Variablen. Ansätze zum Generieren synthetischer Datensätze werden in dieser Arbeit nicht betrachtet, sind jedoch z.B. in Hundepool et al. (2006, S. 89-97) und in Templ (2017, S. 157-177) beschrieben.

3.1 Methoden für kategoriale Variablen

In diesem Unterkapitel werden die Verfahren Recoding, Local Suppression und Post-Randomization Methode (PRAM) näher erläutert.

3.1.1 Recoding

Global Recoding ist eine non-perturbative Methode, "used heavily by statistical offices"(Hundepool et al., 2006, S. 68), bei der die Anzahl an Kategorien reduziert wird, indem die Beobachtungen umkodiert werden. Somit werden weniger detaillierte Informationen vermittelt, was wiederum auch K-Anonymität erreicht. Obwohl am geläufigsten für kategoriale Variablen, kann dieses Verfahren auch für metrische verwendet werden. Während bei den Letztgenannten die Variable diskretisiert wird, werden bei kategorialen Variablen die Kategorien, deren Anzahl an Beobachtungen relativ klein ist, supprimiert bzw. mit anderen Kategorien mit höherer Häufigkeit kombiniert (siehe z.B. Hundepool et al., 2006, S. 68-69; Templ, 2017, S. 100-103). Eine Diskretisierung bei metrischen Variablen führt jedoch "very often to an unaffordable loss of information"(Hundepool et al., 2006, S. 68).

Global Recoding wird nicht nur auf den unsicheren Teil des Datensatzes angewendet, sondern auf den gesamten, "to obtain a uniform categorisation of each variable"(Hundepool et al., 2006, S. 69).

Das Verfahren wird mit folgendem Beispiel in Tabelle 3 dargestellt:

Kategoriale Variable: Land		**Metrische Variable: Alter**	
Original	Umkodiert	Original	Umkodiert
Deutschland	Deutschland	21	20 - 29
Schweiz	Schweiz	35	30 - 39
Deutschland	Deutschland	29	20 - 29
Deutschland	Deutschland	23	20 - 29
Brasilien	Schweiz	36	30 - 39
Schweiz	Schweiz	30	30 - 39

Tabelle 3: Beispiel für Recodingverfahren mit kategorialer und metrischer Variable

Da in den originalen Daten die Kategorie "Brasilien" eine einzige Beobachtung enthält, kann diese mit der Kategorie "Schweiz" z.B. aggregiert bzw. umkodiert werden, um 2- und 3-Anonymität zu erreichen.

Ein Spezialfall der Global Recoding ist die Top und Bottom Coding Methode. "The idea is that top values (those above a certain threshold) are lumped together to form a new category"(Hundepool et al., 2006, S. 69). Dies gilt analog für Bottom Werte, die unterhalb eines bestimmten Thresholds liegen.

In der statistischen Software `R` kann das beschriebene Verfahren mithilfe der Funktionen *globalRecode()* und *TopBotCoding()* aus dem Package `sdcMicro` von Templ et al. (2015) durchgeführt werden.

3.1.2 Local Suppression

Falls nach der oben beschriebenen Umkodierung eindeutige Kombinationen kategorialer Key Variablen verbleiben und somit noch K-Anonymität verletzen, wird in Templ (2017, S. 103) der Einsatz von Local Suppression vorgeschlagen. Ebenso geeignet für kategoriale Variablen werden bei diesem non-perturbativen Verfahren fehlende Werte generiert, welche bestimmte ursprüngliche Werte ersetzen, damit mehr Key Variablen die gleichen Muster teilen, was wiederum das Offenlegungsrisiko reduziert. Local Suppression wird individuell und unabhängig innerhalb einer Beobachtung appliziert, d.h. für jenen Wert in der Kombination der Key Variablen einer bestimmten Beobachtung, nicht aber für alle Beobachtungen.

Es gibt zwei Implementierungsansätze für dieses Verfahren (vgl. Templ, 2017, S. 103-110), die in `R` für Ansätze 1 und 2, jeweils mithilfe der Funktionen *kAnon()* bzw. *LocalSupp()* erfolgen:

1. Versuch K-Anonymität (typischerweise 3-Anonymität) zu erreichen mit minimaler Local Suppression von Werten: Dies wird mittels eines heuristischen Algorithmus durchgeführt, nachdem der User K-Anonymität und die Ordnung der Key Variablen nach der Likelihood für Local Suppression festgelegt hat. Generell gilt dabei, je wichtiger eine Variable ist, desto weniger Local Suppression soll für diese vorgenommen werden.

2. Festlegung eines Risiko-Thresholds auf Beobachtungsebene: Bei dieser Methode wird Local Suppression nur für jene Beobachtungen durchgeführt, deren individuellen Offenlegungsrisiken höher als der festgelegte Threshold sind. Dieser Threshold kann anhand eines mit individuellen Offenlegungsrisiken dargestellten Dichteplots ausgewählt werden.

Einige Algorithmen wurden entwickelt, um heuristische Lösungen für Ansatz 1 zu liefern, nämlich Mondrian-Algorithmus, all-M Ansatz und K-Anonymität (default Ansatz im Package `sdcMicro`). Diese werden ausführlicher in Templ (2017, S. 104) beschrieben.

Für die Darstellung vom Implementierungsansatz 1 in Form eines Beispiels wird erneut die Situation vom Recodingsbeispiel 3 zugrunde gelegt, indem jedoch hier nur 2-Anonymität erreicht wird (siehe Tabelle 4).

Kategoriale Variable: Land		**Metrische Variable: Alter**	
Original	Local Suppression	Original	Umkodiert
Deutschland	Deutschland	21	20 - 29
Schweiz	Schweiz	35	30 - 39
Deutschland	Deutschland	29	20 - 29
Deutschland	Deutschland	23	20 - 29
Brasilien	*	36	30 - 39
Schweiz	Schweiz	30	30 - 39

Tabelle 4: Beispiel für Local Suppression mit kategorialer Variable, indem " * " den generierten fehlenden Wert bezeichnet

Ein weiteres Beispiel für die Illustration des Implementierungsansatzes 2, was die Wahl eines Thresholds anbelangt, wird aus Templ (2017, S. 108) übernommen und ist in Abbildung 3 dargestellt. Aus dieser kann festgestellt werden, dass die meisten Daten weniger als 0.005 Offenlegungsrisiko haben. Dieser Wert kann als Risiko-Threshold dienen und somit werden jene Beobachtungen der abgebildeten Variable, die oberhalb dieses Thresholds liegen, durch fehlende Werte ersetzt.

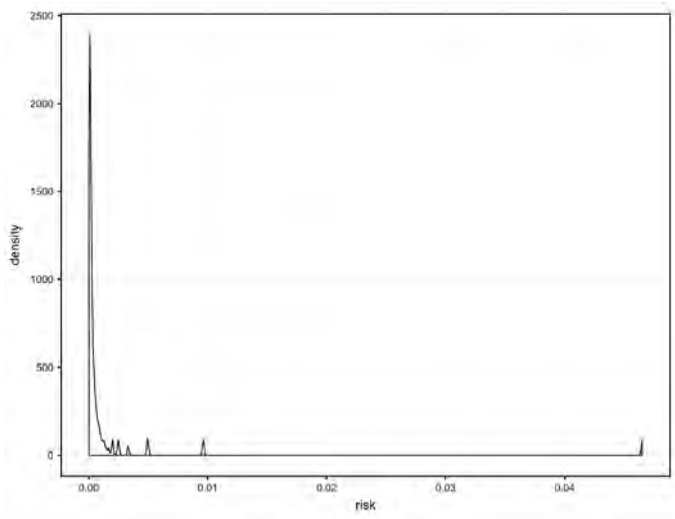


Abbildung 3: Wahl eines Risiko-Thresholds für Local Suppression (aus Templ (2017, S. 108) entnommen)

3.1.3 Post-Randomization Methode (PRAM)

Ein weiteres nun probabilistisch perturbatives Anonymisierungsverfahren für kategoriale Variablen ist die Post-Randomization Methode (PRAM). Im Falle einer großen Anzahl an kategorialen Key Variablen (z.B. mehr als 5) kann Templ (2017, S. 116) zufolge, Recoding das Offenlegungsrisiko nicht ausreichend reduzieren und Local Suppression zu einem allzu großen Informationsverlust führen. Auf Grund dessen mag das PRAM Verfahren

eine effizientere Alternative zu den vorherigen sein. "Originally we developed PRAM as the categorical variable analogon of noise addition to continuous variables"(Gouweleeuw et al., 1998, S. 463). PRAM ist auch mit der Randomised Response (RR) Technik verwandt; beide sind mathematisch äquivalent (van den Hout and van der Heijden, 2002, S. 272). Letztgenannte Methode kommt bei hoch sensiblen Fragen (genauer gesagt: bei hoch sensibler Response Variable) zur Anwendung, deren Beantwortung "face-to-face" wahrscheinlich nicht wahrheitsgemäß ist. Deren latente Häufigkeiten können jedoch aus der Stichprobe der Befragten mittels Einbettung der Fragen in einen Zufallsmechanismus geschätzt werden (vgl. Gouweleeuw et al., 1998, S. 463-464). "PRAM can be seen as applying RR after the data have been collected"(van den Hout and van der Heijden, 2002, S. 270).

Bei PRAM kann für jede Beobachtung (unabhängig von den anderen Beobachtungen) der Score der Kategorien für ausgewählte Variablen vertauscht werden, basierend auf einer vordefinierten quadratischen Transitionsmatrix (auch Markov Matrix genannt (siehe z.B. Hundepool et al., 2006, S. 65; Gouweleeuw et al., 1998, S. 465)), welche die Wahrscheinlichkeiten für jede Kategorie mit anderen vertauscht zu werden angibt (vgl. Gouweleeuw et al., 1998, S. 463; Templ, 2017, S. 116; Hundepool et al., 2006, S. 65-66 & S. 85). Diese Wahrscheinlichkeiten sind (basierend auf z.B. Gouweleeuw et al. (1998, S. 465); Hundepool et al. (2006, S. 86); de Wolf et al. (2002, S. 2)) wie folgt definiert:

$$p_{kl} = P(X^* = l|X = k) \tag{2}$$

wobei X eine kategoriale Variable mit $k = 1, ..., K$ Kategorien im originalen Datensatz, X^* die entsprechende perturbierte Version davon, k und l Scores bzw. Kategorien der kategorialen Variable sind und p_{kl} der Wahrscheinlichkeit entspricht, dass ein ursprünglicher Score $X = k$ in den $X^* = l$ umgewandelt wird. Diese sogenannten Transitionswahrscheinlichkeiten für alle definierten ks gestalten somit die Transitionsmatrix bzw. Markov Matrix $\boldsymbol{P}^{K\times K}$. Dazu muss die Summe der Zeilen in der Transitionsmatrix 1 ergeben. Ferner wird angenommen, dass die Markov Matrix $\boldsymbol{P}$ invertierbar ist, um die wahren Häufigkeiten von X dadurch schätzen zu können (Gouweleeuw et al., 1998, S. 465; van den Hout, 1999, S. 5), was z.B. in (32) geschieht.

Darüber hinaus ist der PRAM Effekt auf eine eindimensionale Häufigkeitstabelle folgendermaßen bestimmt (siehe z.B. Gouweleeuw et al., 1998, S. 469; Hundepool et al., 2006, S. 86; de Wolf, 2006, S. 191):

$$(T_{X^*}|X) = \boldsymbol{P}^t T_X \tag{3}$$

wobei t für transponiert, $T_X = (T_X(1), ..., T_X(K))$ für die originale Häufigkeitstabelle für

die K Kategorien und T_{X^*} für die entsprechende perturbierte Version davon stehen.

Folgendes Beispiel ist von Hundepool et al. (2006, S. 86) sowie von de Wolf (2006, S. 191) inspiriert. Angenommen, die Variable X ist das Geschlecht, mit Scores $X = 1$ für Männer sowie $X = 0$ für Frauen. Die Anwendung von PRAM mit $p_{11} = p_{22} = 0,8$ (d.h. die Wahrscheinlichkeit, dass der Score unverändert bleibt) auf einen Mikrodatensatz mit 90 Männern und 80 Frauen erzeugt perturbierte Daten, mit zu erwartenden Männern gleich 88 ($= 90 \cdot 0{,}8 + 80 \cdot 0{,}2$) sowie 82 ($= 80 \cdot 0{,}8 + 90 \cdot 0{,}2$) Frauen (im Durchschnitt). Vor der Perturbation waren jedoch 16 von den 88 Männern eigentlich Frauen, sowie 18 von den 82 Frauen im originalen Datensatz Männer.

"Basically, it is a form of intended misclassification, using a known and predetermined probability mechanism"(Hundepool et al., 2006, S. 85; vgl. Gouweleeuw et al., 1998, S. 464; de Wolf, 2006, S. 189). Neben der Transitionsmatrix beruht PRAM auch auf einem stochastischen Prozess, der "on the outcome of a random multinomial draw"(Shlomo et al., 2010, S. 42; vgl. Gouweleeuw et al., 1998, S. 469) basiert.

Was bringt eigentlich eine mittels PRAM absichtliche Missklassifikation, wenn Letztgenanntes eher unerwünscht bzw. ungünstig ist?

> *"[...] the risk of identification of respondents is reduced: even in case one could make a link between a record in the microdata file and an individual, the possible incorrectness of the scores yields uncertainty on the correctness of the link."* (de Wolf, 2006, S. 189; de Wolf and van Gelder, 2004, S. 1)

Durch diese "Randomness" bzw. Unsicherheit kann ein Angreifer nie sicher davon ausgehen, dass gegebenfalls die von ihm identifizierte auch die tatsächliche Person ist (vgl. Hundepool et al., 2006, S. 85). Um ein "Mismatch" verursachen zu können, wird vor der Anwendung von PRAM (wie oben angesprochen) eine angemessen zu definierende Transitionsmatrix bzw. Markov Matrix benötigt. Wie kann allerdings eine angemessene Markov Matrix bestimmt werden?

> *"The specific choice of the Markov matrix will influence the disclosure risk. Therefore, the process of choosing the Markov matrix will usually turn out to be an iterative process: when a certain matrix is chosen, the effects on the disclosure risk will be considered. If these effects are not satisfactory, the choice of the matrix should be reconsidered. Then the effects on the disclosure risk should be re-examined."* (de Wolf et al., 2002, S. 2)

Ferner kann PRAM nicht nur auf eine Variable, sondern auf mehrere gleichzeitig verwendet werden, um z.B. "to be able to cope with dependencies between variables"(de Wolf et al., 2002, S. 3). Dadurch können jedoch auch Inkonsistenzen auftreten (da PRAM für

jede Beobachtung unabhängig von den anderen Beobachtungen appliziert wird), nämlich u.a. unwahrscheinliche bzw. unlogische Transitions, wie z.B. ein 5-jähriges verheiratetes Kind oder einen Mann mit Brustkrebs, welche "would attract the attention of a possible intruder"(Hundepool et al., 2006, S. 87; vgl. Gouweleeuw et al., 1998, S. 466), und somit möglichst vermieden werden sollen, indem die entsprechenden Transitionswahrscheinlichkeiten auf 0 gesetzt werden. Weitere Inkonsistenzen und deren mögliche Lösungsvorschläge sind in de Wolf et al. (2002, S. 4-6) diskutiert.

Für die Darstellung dieses Verfahrens kann folgendes zweites Beispiel angenommen werden. Die Variable Land hat drei Kategorien: Land = 1 "Deutschland"(D), Land = 2 "Schweiz"(S), Land = 3 "Brasilien"(B). Des Weiteren wird folgende 3x3 Transitionsmatrix definiert, bei der p_{kl} die Wahrscheinlichkeit darstellt, Kategorie k zu l zu wechseln:

$$\boldsymbol{P} = \begin{pmatrix} & D & S & B \\ D & 0.6 & 0.4 & 0 \\ S & 0.1 & 0.8 & 0.1 \\ B & 0.1 & 0.2 & 0.7 \end{pmatrix}$$

Die Werte 0.6, 0.8 und 0.7 in der Hauptdiagonale der Matrix sind die Wahrscheinlichkeiten, dass die Kategorien in der Variable nach der Perturbation unverändert bleiben. Somit, je höher dieser Wert, desto wahrscheinlicher wird es, dass eine perturbierte Beobachtung der tatsächlichen originalen entspricht. Ferner beträgt die Wahrscheinlichkeit p_{12}, d.h. "Deutschland" zu "Schweiz" zu wechseln, 0.4. Darüber hinaus wurde mit $p_{13} = 0$ als unwahrscheinlich definiert bzw. mathematisch probabilistisch "verboten", "Deutschland" zu "Brasilien" zu wechseln. Dennoch, je höher die Anzahl an 0-Werten in der Transitionsmatrix ist, desto höher mag das Offenlegungsrisiko sein (siehe de Wolf, 2006, S. 201), da die Transitionsmöglichkeiten reduziert werden.

Neben der Verringerung des Offenlegungsrisikos mittels PRAM, können auch Informationen aus den ursprünglichen Daten gewonnen werden, da der verwendete Wahrscheinlichkeitsmechanismus vordefiniert ist und somit, bekannt. All dies führt zu einem wesentlichen Vorteil des Verfahrens, nämlich die Flexibilität der Methode. Dadurch, dass die Transitionsmatrix als Funktionsparameter frei spezifiziert werden kann, können alle beliebigen bzw. gewünschten Effekte modelliert werden (vgl. Templ, 2017, S. 117-118). Darausfolgend, "one can make use of correction methods similar to those used in case of misclassification and randomised response situations"(Hundepool et al., 2006, S. 85; vgl. Gouweleeuw et al., 1998, S. 464), um u.a. die latenten wahren Häufigkeiten aus den perturbierten schätzen zu können. Allerdings besteht noch die Debatte, ob die NSIs Parameter der SDC-Methoden (wie z.B. die Transitionsmatrix) mit den perturbierten Daten mitliefern dürfen aufgrund "the risk of deciphering the perturbation process"(Shlomo et al.,

2010, S. 42).

Mithilfe der Funktion *pram()* kann das Verfahren in `R` angewendet werden. Eine Möglichkeit besteht darin, die Transitionsmatrix selbst zu definieren. Die andere Variante ist, sie mittels eines Zufallsmechanismus zu generieren, indem der Algorithmus die Werte der Variablen (in der Regel die riskanten) zufällig gemäß einer Transitionsmatrix vertauscht. Falls Letztgenanntes zutrifft und Reproduzierbarkeit erwünscht ist, muss ein Seed entsprechend des Zufallsprinzips gesetzt werden.

3.2 Methoden für metrische Variablen

Dieses Unterkapitel beschäftigt sich mit den Verfahren Microaggregation und Noise Addition.

3.2.1 Microaggregation

Die von Defays and Anwar (1998) entwickelte perturbative Anonymisierungsmethode wird üblicherweise für metrische Variablen verwendet. Bei der Anwendung dieser Methode werden die Daten zunächst in Gruppen aufgesplittet und anschließend die individuellen Werte der Beobachtungen für jede definierte Variable durch einen aggregierten Wert ersetzt. "It is also a natural approach to achieving k-anonymity"(Templ, 2017, S. 119), da die gebildete Gruppengröße dem k in Bezug auf K-Anonymität entspricht. Somit folgt: Je größer die Gruppengröße, desto kleiner ist das Offenlegungsrisiko bzw. umso höher ist der Datenschutz. Für den aggregierten Wert wird gängig der Mittelwert genommen; der Median bzw. der Modus stellen jedoch auch Alternativen dar. Die Ersetzung der Werte durch den Mittelwert der Gruppen anstatt anderer Ersatzwerte hat den Vorteil, dass der Overall-Mittelwert der Variablen erhalten bleibt (vgl. Benschop and Welch, 2019).

Wie Templ (2017, S. 119) hervorhebt, "To preserve the multivariate structure of the data, the most challenging part of micro-aggregation is grouping records by how "similar" they are". Ferner betonen Hundepool et al. (2006, S. 77), dass aus der Perspektive des Informationsverlusts der optimale k-Split der ist, welcher die Homogenität innerhalb der Gruppe maximiert, denn je höher Letztgenannte wird, desto geringer der Informationsverlust.

Es gibt diverse Methoden der Microaggregation, welche sich nach Benschop and Welch (2019) bezüglich folgender drei Punkte voneinander unterscheiden: 1) Definition von Homogenität bzw. Ähnlichkeit; 2) Verwendete Algorithmen, die nach der Ähnlichkeit der Gruppen suchen; 3) Bestimmung von Ersatzwerten innerhalb der Gruppen.

Benschop and Welch (2019) schlagen vor, dass bei mehreren für die Microaggregation verwendeten Variablen zunächst die Kovarianz- bzw. Korrelationsmatrix dieser Variablen

betrachtet werden sollte. Wenn nicht alle Variablen, aber zwei oder mehr Gruppen von Variablen eine hohe Korrelation aufweisen, ist der Informationsverlust geringer, wenn die Microaggregation getrennt auf diese Gruppen von Variablen angewendet wird. Der Informationsverlust bei der Anwendung der multivariaten Microaggregation ist generell geringer, wenn die Variablen stark korreliert sind (vgl. Benschop and Welch, 2019). Daher sind korrelierte Variablen innerhalb einer Gruppe, die simultan microaggregiert werden, zu bevorzugen, was auch in (W.E. Winkler, 2004 nach Hundepool et al., 2006, S. 82) betont wird. Diese Aussagen werden in Abschnitt 6.1.4 mit einer Simulation untermauert.

Das Verfahren ist mittels *microaggregation()* in R implementiert. Die Tabelle 5 illustriert dies an einem Beispiel, bei welchem 2-Anonymität erreicht wird:

	Var1	Mic1	Var2	Mic2
1	0.5	0.30	20	15.5
2	1.0	1.10	4	4.5
3	1.2	1.10	5	4.5
4	0.3	1.65	27	40.0
5	3.0	1.65	53	40.0
6	0.1	0.30	11	15.5

Tabelle 5: Beispiel für Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird

Wie kommen die Spalten “Mic1” bzw. “Mic2” zustande? Um diese Werte zu generieren werden drei Schritte durchgeführt:

1. Die Daten werden zunächst im multidimensionalen Raum geordnet und der Index der Ordnung des originalen Datensatzes wird zwischengespeichert.
2. Anschließend werden die Gruppen mit k Elementen jeweils in der geordneten Reihenfolge gebildet bzw. aggregiert, basierend auf einem vordefinierten k für die Erreichung von K-Anonymität (hier $k = 2$).
3. Abschließend wird in jeder Variable der Mittelwert innerhalb jeder Gruppe gebildet, welcher letztendlich den ursprünglichen Wert ersetzt. Als Letztes werden die Daten entsprechend des zwischengespeicherten Index angeordnet, um die originale Reihenfolge beizubehalten.

	Var1	Var2
1	0.5	20
6	0.1	11
2	1.0	4
3	1.2	5
4	0.3	27
5	3.0	53

⇒

	Mic1	Mic2
1	0.3	15.5
6	0.3	15.5
2	1.10	4.5
3	1.10	4.5
4	1.65	40.0
5	1.65	40.0

⇒

	Mic1	Mic2
1	0.3	15.5
2	1.10	4.5
3	1.10	4.5
4	1.65	40.0
5	1.65	40.0
6	0.3	15.5

Im Folgenden werden die *Individual Ranking*-, *MDAV*-, *RMD*-, *PCA*- und *Simple* Methoden näher erläutert.

3.2.1.1 *Individual Ranking* Methode

Die simpelste Methode ist die von Defays and Anwar (1998) entwickelte univariate Microaggregation, auch *Individual Ranking* genannt (in `R` als Methode *onedims* implementiert). Bei dieser werden die Werte zunächst spaltenweise geordnet und innerhalb einer Gruppe durch die spaltenweise unabhängig aggregierten Werte ersetzt, analog zu den Schritten in 3.2.1, jedoch nur eindimensional, d.h. eine Spalte nach der anderen.

Ein angemessenes Kriterium zur Messung der Homogenität bei dieser univariaten Methode ist das Quadratsummenkriterium, welches folgendermaßen definiert ist:

$$SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)^T (x_{ij} - \bar{x}_i)$$

Da "The lower SSE, the higher the within group homogeneity"(Hundepool et al., 2006, S. 77), muss der Sum of Squares Error (SSE) minimiert werden, damit der optimale k-Split erreicht wird.

Dadurch, dass die Veränderungen in den Variablen begrenzt sind, führt dies zu einem minimalen vorteilhaften Informationsverlust (Benschop and Welch, 2019). Als Nachteil hat sich jedoch (in Domingo-Ferrer et al., 2002 nach Benschop and Welch, 2019) feststellen lassen, dass das Offenlegungsrisiko sehr hoch sein kann, wenn diese Methode auf mehrere Variablen angewendet und kein weiteres Anonymisierungsverfahren durchgeführt wird.

Um dies an einem Beispiel zu zeigen wird erneut auf die erste Situation 5 zurückgegriffen, in welcher Var1 und Var2 gegeben sind (siehe Tabelle 6):

	Var1	Mic1	Var2	Mic2
1	0.5	0.75	20	15.5
2	1.0	0.75	4	4.5
3	1.2	2.10	5	4.5
4	0.3	0.20	27	40.0
5	3.0	2.10	53	40.0
6	0.1	0.20	11	15.5

Tabelle 6: Beispiel für Methode *onedims* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird

3.2.1.2 *MDAV* Methode

Der Standard für Microaggregation im Package `sdcMicro` (Templ et al., 2015) ist die Maximum Distance to Average ($MDAV$) Methode, welche die Daten basierend auf der

klassischen euklidischen Distanz im multidimensionalen Raum aggregiert. Im Beispiel 5 kommt genau diese zur Anwendung.

Diese Art von Microaggregation (als *mdav* in `R` implementiert) kann in folgende Schritte aufgesplittet werden (vgl. Templ, 2017, S. 120-123):

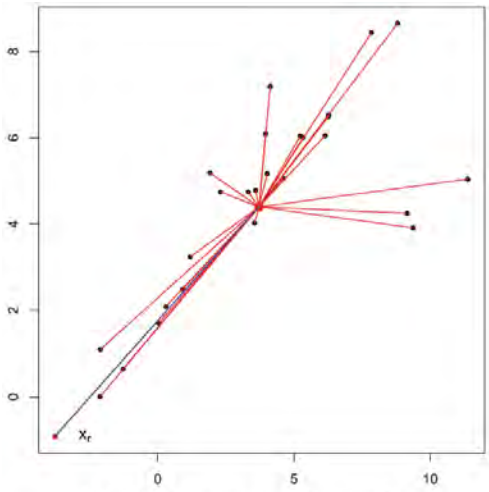
1. Das Zentrum der Daten wird mittels spaltenweisen Durchschnitts geschätzt.
2. Die am weitesten (basierend auf der euklidischen Distanz) vom Zentrum entfernte Beobachtung (angenommen $\mathbf{x_r}$) wird genommen.
3. Die am weitesten (basierend auf der euklidischen Distanz) von $\mathbf{x_r}$ entfernte Beobachtung wird ebenso genommen (z.B. $\mathbf{x_s}$).
4. k - 1 nächste Nachbarn werden jeweils mittels euklidischer Distanz für $\mathbf{x_r}$ und $\mathbf{x_s}$ selektiert.
5. Anschließend wird spaltenweise der Durchschnitt für beide Gruppen berechnet, der dann die Werte der entsprechenden Beobachtungen in jeder Gruppe ersetzt.
6. Die Schritte 1 bis 5 werden solange wiederholt, bis maximal 2k - 1 Beobachtungen übrig sind, welche dann zusammen aggregiert werden.

Der ausführliche Ablauf dieser Methode kann Templ (2017, S. 122) entnommen werden, und ist in Abbildung 4 dargestellt.

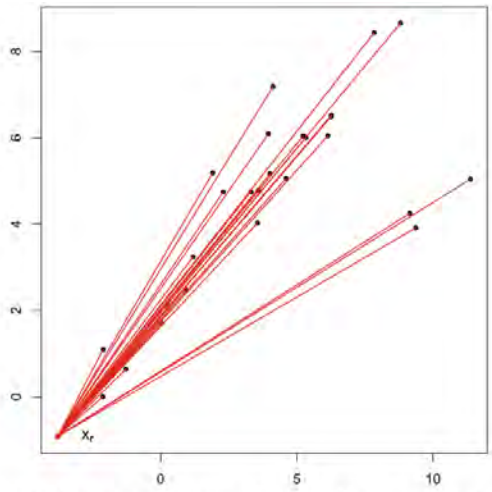
3.2.1.3 *RMD* Methode

Die Erweiterung der vorherigen Methode ersetzt die euklidische durch die Robust Mahalanobis Distanz (RMD) und die "multivariate data are dealt with by taking the (robust) covariance structure of the data into account"(Templ and Meindl, 2008, S. 117). In Templ (2007, S. 6-7) wird die ganze Prozedur für RMD wie folgt von $MDAV$ adaptiert:

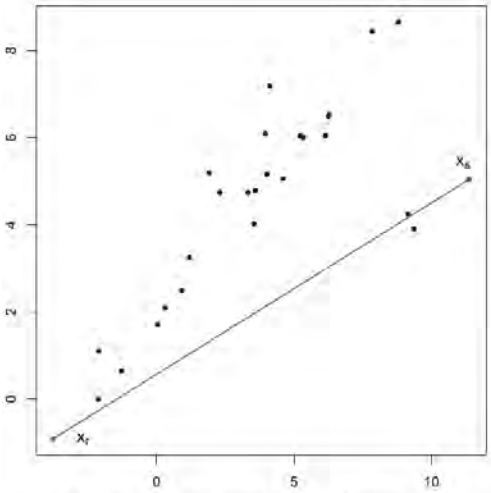
1. Das robuste Zentrum der Daten (d.h. der L1-Median bzw. der koordinatenweise Median) wird berechnet.
2. Die am weitesten (basierend auf der robusten Mahalanobis Distanz) vom robusten Zentrum entfernte Beobachtung (angenommen $\mathbf{x_r}$) wird genommen. Die für die Berechnung der robusten Mahalanobis Distanz benötigte robuste Kovarianzmatrix kann mittels Minimum Covariance Determinant (MCD) Schätzers geschätzt werden.
3. Die am weitesten (basierend auf RMD) von $\mathbf{x_r}$ entfernte Beobachtung wird ebenso genommen (z.B. $\mathbf{x_s}$).
4. k - 1 nächste Nachbarn werden mittels RMD für $\mathbf{x_r}$ und $\mathbf{x_s}$ jeweils selektiert, deren Werte anschließend durch einen aggregierten Wert (z.B. arithmetisches Mittel) in jeder Gruppe ersetzt werden.



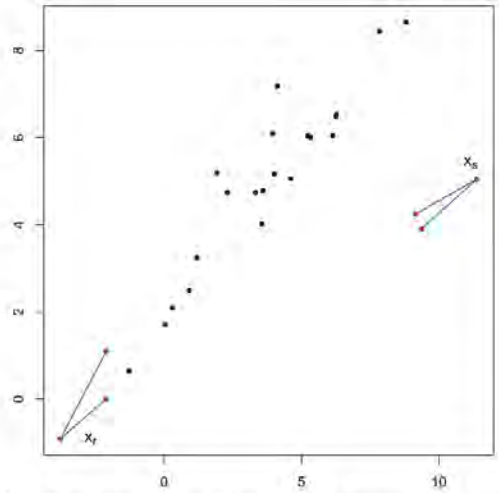
(a) Distances to the center.



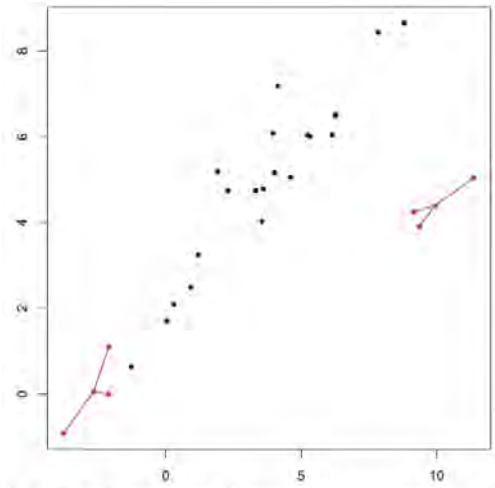
(b) Distances from observation $\mathbf{x}_r$ that is farthest from center to all other observations.



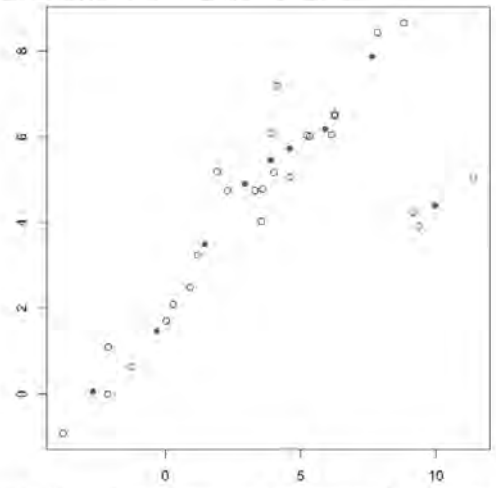
(c) Farthest point $\mathbf{x}_s$ from $\mathbf{x}_r$.



(d) Determining the $k-1$ (here: 2) nearest neighbors of $\mathbf{x}_r$ and $\mathbf{x}_s$.



(e) Replacement/aggregation with the mean.



(f) Resulting microaggregated observations (in blue) after applying the same procedure as long as all observations are microaggregated.

Abbildung 4: Ablauf *MDAV* Methode für Microaggregation (aus Templ (2017, S. 122) entnommen)

5. Der ursprüngliche Datensatz, minus die aggregierten Daten aus dem letzten Schritt, wird als neuer Datensatz genommen und alle Schritte erneut durchgeführt, bis alle Daten microaggregiert sind.

3.2.1.4 *Gower Distance* Methode

Eine weitere Methode basierend auf Distanzen, die sowohl auf kategoriale als auch auf metrische Variablen (also gemischte Daten) gleichzeitig angewendet werden kann, ist die *Gower Distance* Methode. Diese ist aber "Not very commonly used"(Templ, 2017, S. 124), weswegen hier auf die Erklärung verzichtet wird. Nähere Details sind in Templ (2017, S. 123-124 & Unterkapitel 5.2.1) zu finden.

3.2.1.5 *PCA* Methode

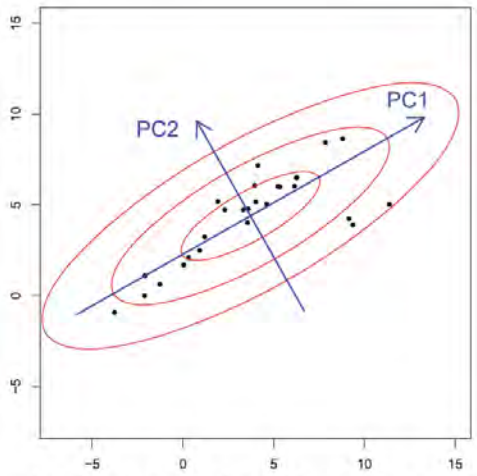
Dies ist eine Projektionsmethode, in der die Daten, basierend auf der Hauptkomponentenanalyse (PCA), nach der ersten Hauptkomponente sortiert werden (als *pca* Methode in `R` implementiert). Erklärt die erste Hauptkomponente einen hohen Prozentsatz der Varianz der für die Microaggregation definierten Variablen, ist die PCA Methode schnell und hat eine gute Performance. Ist das nicht der Fall, sind Templ (2017, S. 120) zufolge andere Methoden bzw. Algorithmen zu bevorzugen.

Var1 und Var2 aus Tabelle 5 kommen erneut zur Anwendung, um die PCA Methode in einem Beispiel in Tabelle 7 darzustellen:

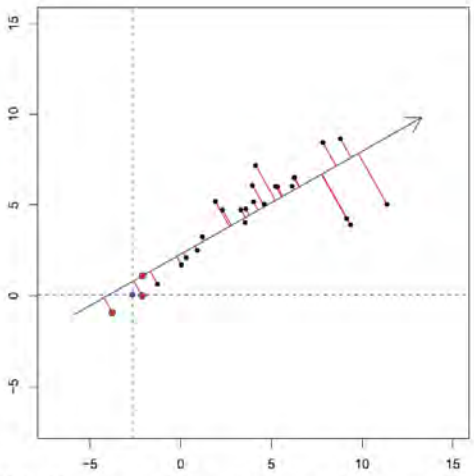
	Var1	Mic1	Var2	Mic2
1	0.5	0.85	20	12.5
2	1.0	0.55	4	7.5
3	1.2	0.85	5	12.5
4	0.3	1.65	27	40.0
5	3.0	1.65	53	40.0
6	0.1	0.55	11	7.5

Tabelle 7: Beispiel für Methode *pca* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird

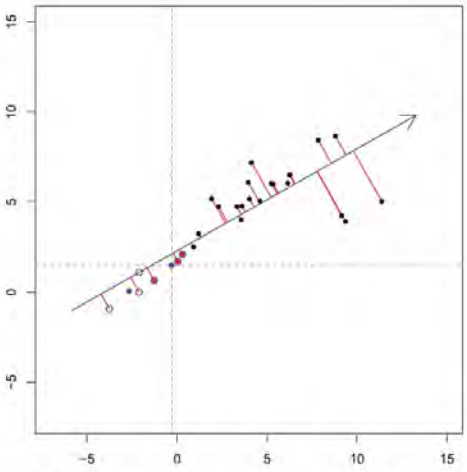
Der detaillierte Ablauf der PCA Methode wird in Abbildung 5 bei zweidimensionalen Beispiel-Daten illustriert, welche ebenso aus Templ (2017, S. 121) entnommen wurde. Zunächst wird die erste Hauptkomponente geschätzt, entlang deren werden die Werte anschließend aggregiert.



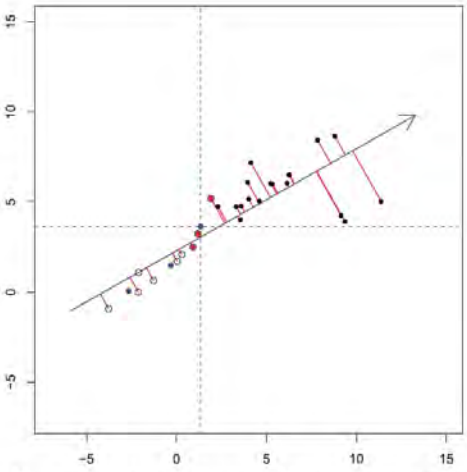
(a) First two principal components.



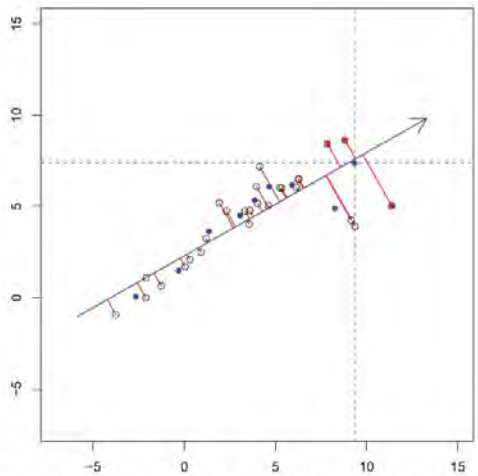
(b) Projection/sorting on first component. Aggregation along the sorted values (first three observations).



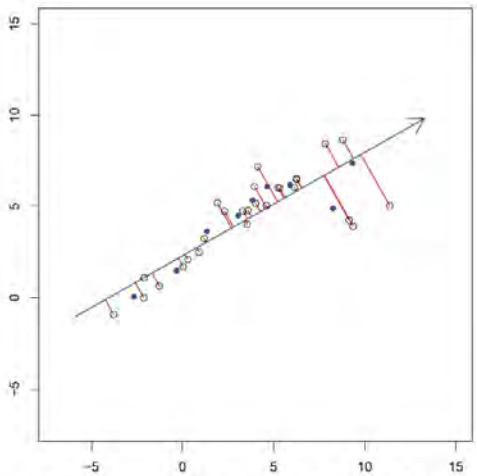
(c) Aggregation along the sorted values (next three observations).



(d) Aggregation along the sorted values (next three observations).



(e) Aggregation along the sorted values (remaining last observations).



(f) Microaggregated values in blue.

Abbildung 5: Ablauf *PCA* Methode für Microaggregation (aus Templ (2017, S. 121) entnommen)

Diese Methode ist dennoch sehr empfindlich gegenüber Outliers, da die Eigenwerte und Eigenvektoren aus einer non-robusten Kovarianzmatrix berechnet werden, was zu ungünstigen Ergebnissen bei der Sortierung nach der ersten Hauptkomponente führen kann (Templ, 2006, S. 350). Eine robuste Version (*pppca* Methode - Sortierung mit Projektion Verfolgung ("Pursuit") von PCA) kann stattdessen verwendet werden. Anstelle einer robusten Schätzung der Varianz bzw. Kovarianz wird bei dieser Methode nach "directions with maximal variance of the data projected on it" (Templ, 2006, S. 351) gesucht, indem ein robuster Skalenschätzer S_n (Projection Pursuit Index) in folgender Formel zum Einsatz kommt, welche den ersten Eigenvektor definiert (idem):

$$v_{S_n,1} = \underset{||a||=1}{\operatorname{argmax}} S_n(a^t x_1, ..., a^t x_n)$$

mit $x_1, ..., x_n$ die Beobachtungen $\in \mathbb{R}^p$ (p die Anzahl der Variablen).

Anschließend wird der Eigenwert wie folgt definiert:

$$\lambda_{S_n,1} = S_n^2((v_{S_n,1})^t x_1, ..., (v_{S_n,1})^t x_n)$$

Templ (2006, S. 351) schlägt den Median Absolute Deviation (MAD) für den Skalenschätzer S_n als Alternative zur klassischen Standardabweichung vor:

$$\text{MAD} = 1.4826 \cdot median(|x_i - median(X)|)$$

wobei x_i für eine Beobachtung und $median(X)$ für den Median der Daten stehen. Zudem sorgt der Wert 1.4826 für Konsistenz bei Normalverteilungen (idem).

Eine weitere Version (*clustpppca* Methode) wurde entwickelt, in der die Sortierung mit Projektion Verfolgung von PCA auf geclusterte Daten angewendet wird. Bei dieser letztgenannten Version ist eine robuste Schätzung der Kovarianzmatrix notwendig, um die erste Hauptkomponente robust zu schätzen, was jedoch nur für kleine oder mittelgroße Datensätze möglich ist, weswegen sie nur in diesen Fällen anwendbar ist (vgl. Templ and Meindl, 2008, S. 116-117; Templ, 2017, S. 120).

Templ (2006, S. 355) stellt noch zwei Plots als Beispiel zur Verfügung, bei denen die Kovarianzstruktur der *pca*- mit der *clustpppca* Methode verglichen werden können (siehe Abbildung 6). Dabei sind die Kovarianzen der originalen Daten in Schwarz und die mit der jeweiligen Methode der Microaggregation perturbierten Daten in Blau abgebildet. Während die *pca* Version keine optimalen Ergebnisse liefert, fällt die *clustpppca* Methode angemessener aus.

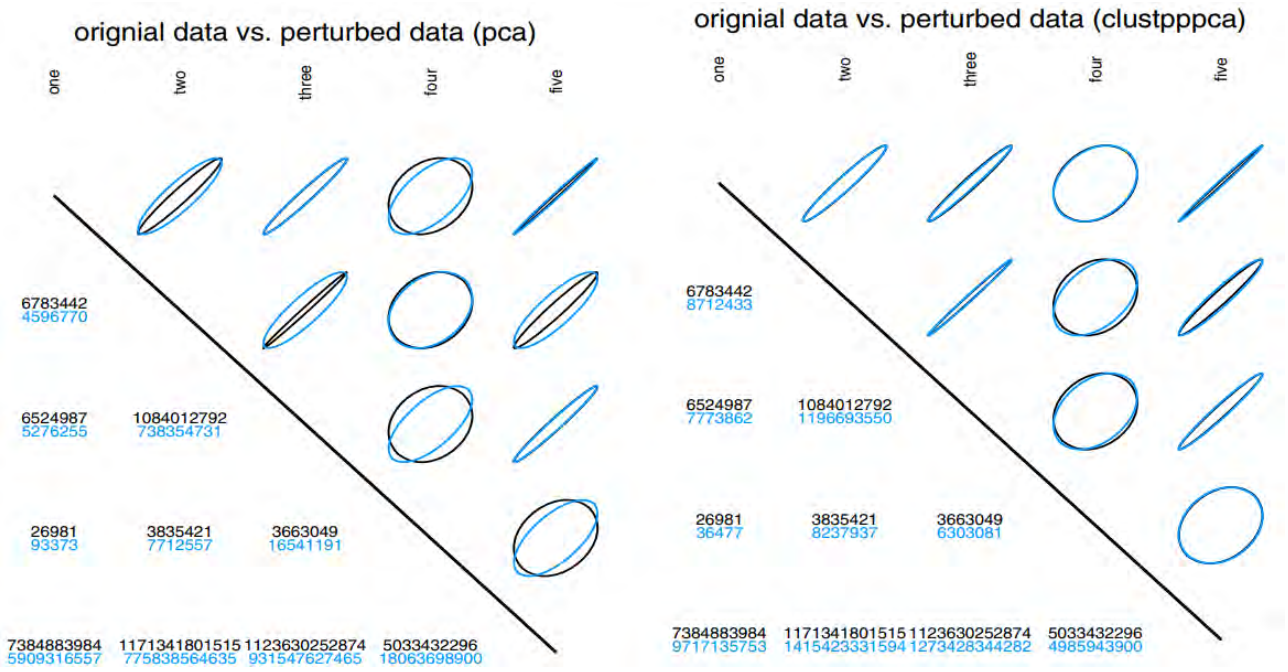


Abbildung 6: Vergleich der Kovarianzen zwischen *pca* und *clustpppca* Methoden (in Blau) mit den (in Schwarz) originalen Daten (aus Templ (2006, S. 355) entnommen)

3.2.1.6 *Simple* Methode

Anders als für die vorherigen Methoden werden hier die Daten vor der Microaggregation nicht sortiert. Diese Methode soll als Benchmark dienen, d.h. einen Vergleich dazu darstellen, inwiefern die Sortierung der Daten vor der Aggregation zu einer Verbesserung führt (Templ et al., 2015).

Die Anwendung dieser Methode auf die Ausgangssituation mit Var1 und Var2 in 5 wird in Tabelle 8 illustriert:

	Var1	Mic1	Var2	Mic2
1	0.5	0.75	20	12
2	1.0	0.75	4	12
3	1.2	0.75	5	16
4	0.3	0.75	27	16
5	3.0	1.55	53	32
6	0.1	1.55	11	32

Tabelle 8: Beispiel für Methode *simple* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird

3.2.2 Noise Addition

Ein weiteres gängiges Anonymisierungsverfahren für metrische Variablen, neben Microaggregation, ist Noise (Rauschen) Addition. Die Art und Weise, wie die Daten mit Noise perturbiert werden, hängt sowohl von der Methode als auch von der Menge an Noise ab. Die Methoden *additive*, zwei Versionen von *correlated* sowie *outdect* (für Outliers) wurden u.a. von Templ et al. (2015) entwickelt und im Package `sdcMicro` implementiert. Eine *multiplikative* Noise Methode ist im Package nicht enthalten.

Beide Hyperparameter (Methode und Noise) können u.a. in der in `R` implementierten Funktion *addNoise()* festgelegt werden. Ist Reproduzierbarkeit der Werte erwünscht, sollte ein Seed gesetzt werden, da die Werte in den folgenden Methoden, anders als bei den bisher beschriebenen Verfahren, aus einem Zufallsmechanismus stammen.

Auf die Methoden *additive*, *correlated*, *correlated*2 und *outdect* wird im Folgenden umfangreicher eingegangen. Weitere Methoden wie z.B. *restr* und Random Orthogonal Matrix Masking (*ROMM*) werden hier nicht näher beschrieben, können aber aus Templ (2017, Kapitel 4) entnommen werden.

3.2.2.1 *Additive* Methode

Basierend u.a. auf Templ (2017, S. 125) kann *additive* unkorrelierte Noise wie folgt formuliert werden:

$$X_j^* = X_j + \varepsilon_j \tag{4}$$

wobei X_j die originalen Werte einer j-ten Variable, X_j^* die perturbierten Werte und ε_j die von X unabhängige unkorrelierte additive Noise ($X \perp \varepsilon$) bezeichnen. Darüber hinaus gilt $X \sim (\mu, \Sigma_X)$, $\varepsilon \sim N(0, \alpha \cdot diag(\sigma_1^2, \sigma_2^2, ... , \sigma_p^2))$, $Cov(\varepsilon_i, \varepsilon_j) = 0$. Somit $X^* \sim (\mu, \Sigma_{X^*})$, wobei $\Sigma_{X^*} = \Sigma_X + \alpha \cdot diag(\sigma_1^2, \sigma_2^2, ... , \sigma_p^2)$, mit α die konstante zu addierende Menge an Noise > 0 und p die Anzahl der Variablen sind (vgl. Brand and Giessing, 2002, S. 3-4; Templ, 2017, S. 125; Hundepool et al., 2006, S. 73).

Dabei bleiben die Mittelwerte und Kovarianzen (im engeren Sinne) erhalten, während die Varianzen bzw. Korrelationsmatrizen nicht erhalten bleiben. Im Folgenden werden diese Aussagen bewiesen:

- $\mathbb{E}(X^*) = \mathbb{E}(X) + \mathbb{E}(\varepsilon) = \mathbb{E}(X) + 0 = \mathbb{E}(X) = \mu$
- $Var(X^*) = Var(X) + Var(\varepsilon) = Var(X) + \alpha \cdot Var(X) = \sigma^2 + \alpha \cdot \sigma^2 = (1 + \alpha)\sigma^2$

- $Cov(X_i^*, X_j^*) = Cov(X_i, X_j)\ \forall\ i \neq j$:

$$\Sigma_{X^*} = \Sigma_X + \Sigma_\varepsilon = \begin{bmatrix} \sigma_{11}^2 & \dots & \sigma_{1p}^2 \\ \vdots & \ddots & \vdots \\ \sigma_{n1}^2 & \dots & \sigma_{np}^2 \end{bmatrix} + \alpha \begin{bmatrix} \sigma_{11}^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_{np}^2 \end{bmatrix} = \begin{bmatrix} (1+\alpha)\sigma_{11}^2 & \dots & \sigma_{1p}^2 \\ \vdots & \ddots & \vdots \\ \sigma_{n1}^2 & \dots & (1+\alpha)\sigma_{np}^2 \end{bmatrix}$$

- $\rho(x_i^*, x_j^*) = \frac{Cov(X_i^*, X_j^*)}{\sqrt{Var(X_i)Var(X_j)}} = \frac{1}{1+\alpha}\rho(x_i, x_j)\ \forall\ i \neq j$

Auf die ursprünglichen Var1 und Var2 aus 5 wird *additive* Noise mit Seed 2022 und Menge an Noise gleich 50 hinzugefügt, wie in Tabelle 9 illustriert. Dabei werden die neuen Werte mittels additiver Noise Perturbation aus dem obigen Zufallsmechanismus generiert. Dabei spielt aber auch der Stichprobenumfang eine wichtige Rolle; je größer er wird, desto präziser werden die Aussagen in 3.2.2.1.

	Var1	addNoise1	Var2	addNoise2
1	0.5	0.98	20	10.23
2	1.0	0.38	4	6.56
3	1.2	0.73	5	11.91
4	0.3	-0.46	27	29.23
5	3.0	2.83	53	62.28
6	0.1	-1.43	11	9.29

Tabelle 9: Beispiel für Methode *additive* in Noise Addition mit metrischen Variablen

Templ (2017, S. 126) illustriert mittels eines Plots den angesprochenen Erhalt der Kovarianzen (im engeren Sinne) bei Verwendung von additiver Noise. Zudem wird eine 97,5% Toleranz Ellipse für die Kovarianzen sowohl der ursprünglichen als auch der perturbierten Daten gezeichnet. Dies wird in Abbildung 7 dargestellt.

Was die Varianzstruktur vor und nach additiver Noise anbelangt, bringen Benschop and Welch (2019) ein weiteres Beipiel mittels eines Plots. Dieser zeigt, wie der additive Noise Effekt einer metrischen Variable aussieht (siehe Abbildung 8). Die dargestellten Histogramme zeigen deutlich, dass, je mehr *additive* Noise auf die originalen Daten addiert wird, desto ferner wird die Verteilung von den ursprünglichen Werten und umso näher an eine Normalverteilung. Des Weiteren bleibt der Mittelwert präserviert, die Varianz nimmt jedoch, mit zunehmender Menge an Noise, zu und mit einem Noiseniveau von 5 ist die Verteilung komplett zerstört (vgl. Benschop and Welch, 2019).

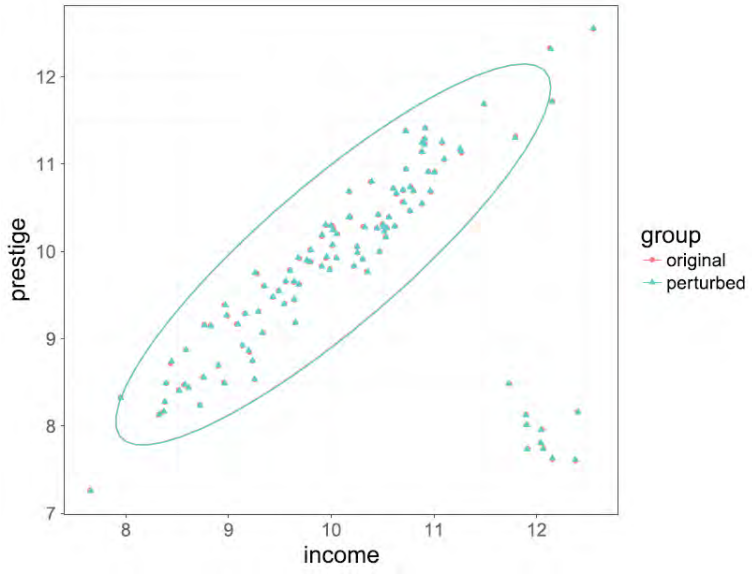


Abbildung 7: Gleiche Kovarianzen (im engeren Sinne) zwischen originalen und perturbierten Daten mit additiver Noise Perturbation (aus Templ (2017, S. 126) entnommen)

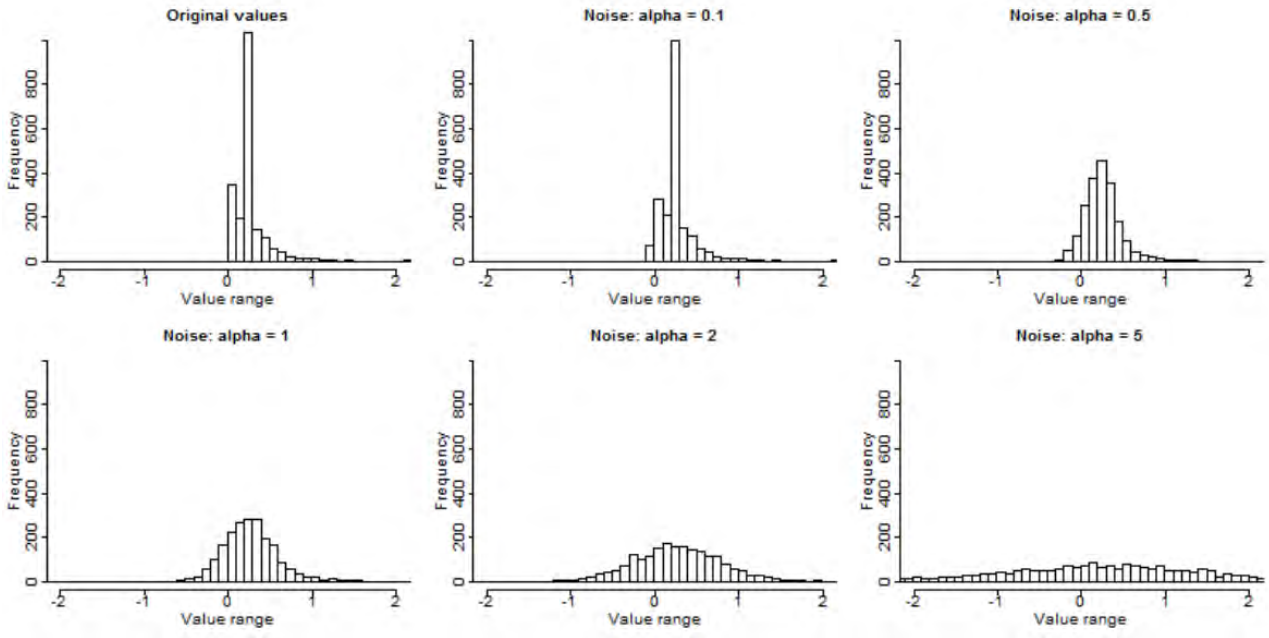


Abbildung 8: Frequenzverteilung einer metrischen Variable vor und nach additiver Noise in Bezug auf die Varianzstruktur (aus Benschop and Welch (2019) entnommen)

Benschop and Welch (2019) stellen daneben mithilfe eines weiteren Plots fest, dass, während der Wertebereich sich mit zunehmender Menge an additiver Noise vergrößert, der Median in etwa unverändert bleibt, hier in Abbildung 9 zu sehen. Dabei sind das Minimum, das 20., 30. und 40. Perzentil, der Median (mit rotem "+" -Symbol gekennzeichnet),

das 60., 70., 80. und 90. Perzentil sowie das Maximum dargestellt.

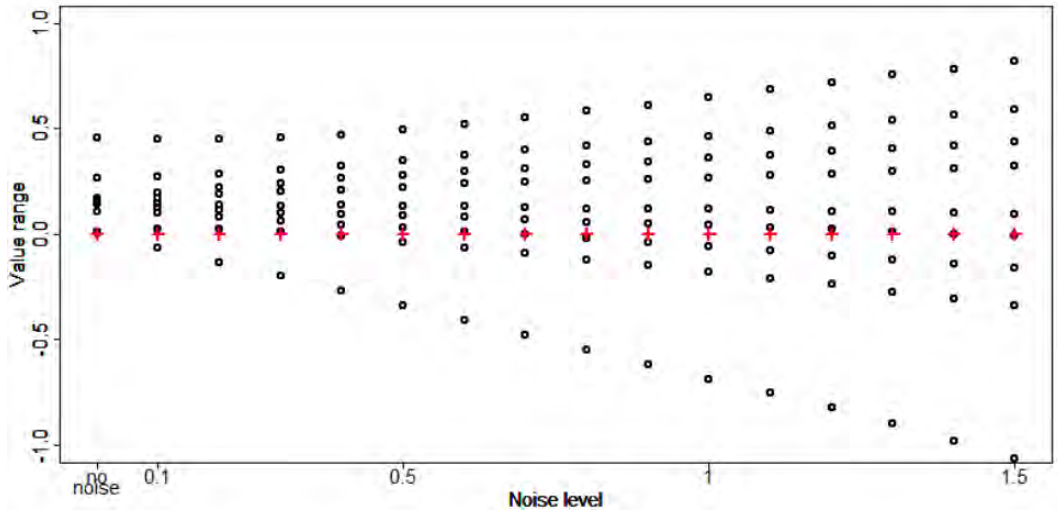


Abbildung 9: Menge an Noise und deren Auswirkungen auf den Wertebereich (Perzentile)(aus Benschop and Welch (2019) entnommen)

Brand and Giessing (2002, S. 3) heben hervor: "it is often implicitly assumed that the errors generated for different variables are independent". Dies hat, wie bereits gesehen, zur Folge, dass die Varianzen um den Faktor $(1 + \alpha)$ "inflated" werden. "This implies that the sample variances of the masked data are asymptotically biased estimators for the variances of the original data"(Brand and Giessing, 2002, S. 4).

Ein weiterer Nachteil dieser additiven Methode ist das Auftreten negativer Werte, was auch schon in der letzten Abbildung ersichtlich war. Das ist besonders problematisch bei Variablen wie z.B. Einkommen oder Kosten, deren Werte per se nicht negativ sein können. Eine Lösung dazu ist die Anwendung von multiplikativer anstatt additiver Noise, wie in Benschop and Welch (2019) vorgeschlagen.

3.2.2.2 *Correlated* Methode

Ein bereits erwähntes Manko der obigen Methode ist, dass die Korrelationsmatrizen nicht erhalten bleiben. Auf Grund dessen ist es optimaler, die Daten mittels korrelierter Noise zu perturbieren, um proportionale Werte bezüglich ursprünglicher Kovarianz- bzw. Korrelationsmatrix zu erhalten (vgl. Templ, 2017, S. 127; Brand and Giessing, 2002, S. 4).

Der Unterschied zur *additive* Noise Methode liegt somit bei $Cov(\varepsilon_i, \varepsilon_j)$, welche im vorherigen Teil Null aufwies (siehe 3.2.2.1), während sie nun proportional zu den ursprünglichen Daten wird (d.h. $Cov(\varepsilon_i, \varepsilon_j) = \alpha \cdot Cov(x_i, x_j)$), indem $\varepsilon \sim N(0, \Sigma_\varepsilon = \alpha \cdot \Sigma_X)$. Zudem gilt weiterhin $X \perp \varepsilon$ sowie auch Folgendes (vgl. Templ and Meindl, 2008, S.

114; Brand and Giessing, 2002, S. 4; Templ, 2017, S. 127; Hundepool et al., 2006, S. 73):

$$\Sigma_{X^*} = \Sigma_X + \Sigma_\varepsilon = \Sigma_X + \alpha \cdot \Sigma_X = (1 + \alpha)\Sigma_X \tag{5}$$

Die Korrelationsstruktur wird in Hundepool et al. (2006, S. 74) wie folgt definiert:

$$\rho(x_i^*, x_j^*) = \frac{1+\alpha}{1+\alpha} \frac{Cov(X_i^*, X_j^*)}{\sqrt{Var(X_i)Var(X_j)}} = \rho(x_i, x_j)$$

Daraus folgt, wie Brand and Giessing (2002, S. 4) betonen, dass sich alle Werte in der Kovarianzmatrix der perturbierten Daten von den originalen um den Faktor $(1 + \alpha)$ unterscheiden. Somit

> *"[...] correlations of the original data can be estimated asymptotically unbiased by the sample correlations of the masked data (see e.g. Kim 1986, 1990). Kim (1990) shows that expected values and covariances of subpopulations can be estimated consistent as long as α is known, too. Furthermore it is possible to obtain consistent estimates of these parameters when only some of the variables used are masked (partial masks, see Kim (1990)."* (idem)

Nun wird *correlated* Noise mit Seed 2022 und Menge an Noise gleich 50 auf die bekannten Var1 und Var2 addiert, was in Tabelle 10 dargestellt wird:

	Var1	corNoise1	Var2	corNoise2
1	0.5	1.54	20	31.71
2	1.0	0.25	4	-11.29
3	1.2	0.33	5	-6.69
4	0.3	-0.57	27	8.17
5	3.0	2.28	53	48.71
6	0.1	-1.27	11	-26.82

Tabelle 10: Beispiel für Methode *correlated* in Noise Addition mit metrischen Variablen

Templ (2017, S. 127) illustriert analog zur Abbildung 7 mittels eines Plots ein Beispiel, wie die Kovarianzstruktur mit *correlated* Noise nun proportional zu den originalen Daten ist. Zudem wird eine 97,5% Toleranz Ellipse für die Kovarianzstruktur, sowohl der ursprünglichen, als auch der perturbierten Daten gezeichnet. Dies wird in Abbildung 10 dargestellt.

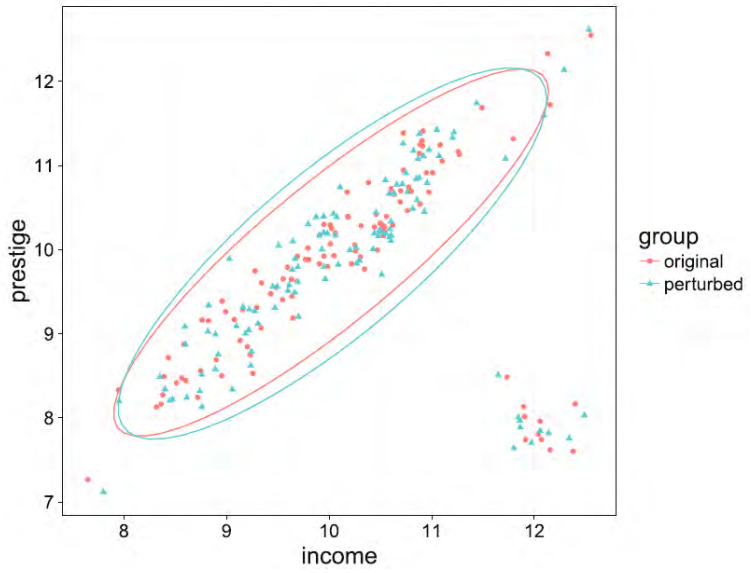


Abbildung 10: Proportionale Kovarianzstruktur zu originalen Daten mit *correlated* Noise Perturbation, mit der Normalverteilungsannahme (aus Templ (2017, S. 127) entnommen)

Ferner nimmt die *correlated* Methode an, die zu perturbierenden Variablen seien annähernd normalverteilt (siehe z.B. Benschop and Welch, 2019; Templ et al., 2015, S. 26). Dies kann u.a. mit einem Jarque-Bera bzw. Shapiro-Wilk Test in `R` untersucht werden, welche z.B. mittels Funktion *normalTest()* und deren entsprechenden Methode (d.h. Kolmogorov-Smirnov Test, Shapiro-Wilk Test oder Jarque-Bera Test) aus dem Package `fBasics` von Wuertz et al. (2022) angegeben werden können.

3.2.2.3 *Correlated2* Methode

Sind die zu perturbierenden Variablen nicht normalverteilt wie in *correlated* Methode angesprochen, muss nach einer Alternative gesucht werden. Dies führt zu einer adjustierten Version der *correlated* Noise basierend auf Transformationen, nämlich die von Kim (1986) entwickelte *correlated2* Methode, welche robust gegen die Normalitätsannahme ist (Benschop and Welch, 2019).

Was die Beschreibung dieser Methode anbelangt, gibt es einen Unterschied zwischen Templ (2017, S. 128), Templ and Meindl (2008, S. 115) und Cano and Torra (2011, S. 7) dahingehend, dass bei den zwei Erstgenannten ein zusätzliches ε in der Definition von d (siehe unten) auftaucht. Diese Arbeit stützt sich allerdings auf die letzte Quelle, da sie plausibler erscheint.

Basierend auf Cano and Torra (2011, S. 7) wird somit hier $d = \sqrt{(1-\alpha^2)}$ berechnet, wobei α die zu addierende Menge an unkorrelierter additiven Noise (zwischen 0 und 1) bezeichnet. Darauffolgend wird $X_j^* = X_j d + \alpha\varepsilon_j$ berechnet, wobei X_j die originalen

Werte einer j-ten Variable, X_j^* die perturbierte Version davon und ε_j zufällige gezogene Zahlen aus einer Normalverteilung $N(\mu_j^* = \frac{\mu_j(1-d)}{\alpha}, Var_j^* = \sigma_j^2)$ bezeichnen. Ferner gilt weiterhin $X \perp \varepsilon$. Die ganze Prozedur führt dazu, dass $\mathbb{E}(X_j^*) = \mathbb{E}(X_j)$ sowie $Var(X_j^*) = Var(X_j)$ sind (Cano and Torra, 2011, S. 7). Im Folgenden folgt noch der Beweis dazu:

$$
\begin{aligned}
\mathbb{E}(X_j^*) &= \mathbb{E}(dX_j + \alpha\varepsilon_j) \\
&= d\,\mathbb{E}(X_j) + \alpha\,\mathbb{E}(\varepsilon) \\
&= d\mu_j + \alpha\frac{\mu_j(1-d)}{\alpha} \\
&= d\mu_j + \mu_j - d\mu_j \\
&= \mu_j = \mathbb{E}(X_j)
\end{aligned}
\qquad
\begin{aligned}
Var(X_j^*) &= Var(dX_j + \alpha\varepsilon_j) \\
&= d^2 Var(X_j) + \alpha^2 Var(\varepsilon_j) \\
&= (1-\alpha^2)\sigma_j^2 + \alpha^2\sigma_j^2 \\
&= \sigma_j^2 - \alpha^2\sigma_j^2 + \alpha^2\sigma_j^2 \\
&= \sigma_j^2 = Var(X_j)
\end{aligned}
$$

Nun wird die *correlated2* Noise Methode mit Seed 2022 und $\alpha = 0.1$ auf die bekannten Var1 und Var2 addiert:

	Var1	cor2Noise1	Var2	cor2Noise2
1	0.5	0.60	20	18.05
2	1.0	0.88	4	4.59
3	1.2	1.10	5	6.46
4	0.3	0.15	27	27.41
5	3.0	2.96	53	54.69
6	0.1	-0.20	11	10.70

Tabelle 11: Beispiel für Methode *correlated2* in Noise Addition mit metrischen Variablen

Ferner zeigt die Templ (2017, S. 128) entnommene Abbildung 11 die *correlated2* Noise Variante des Plots zu 10 und 7:

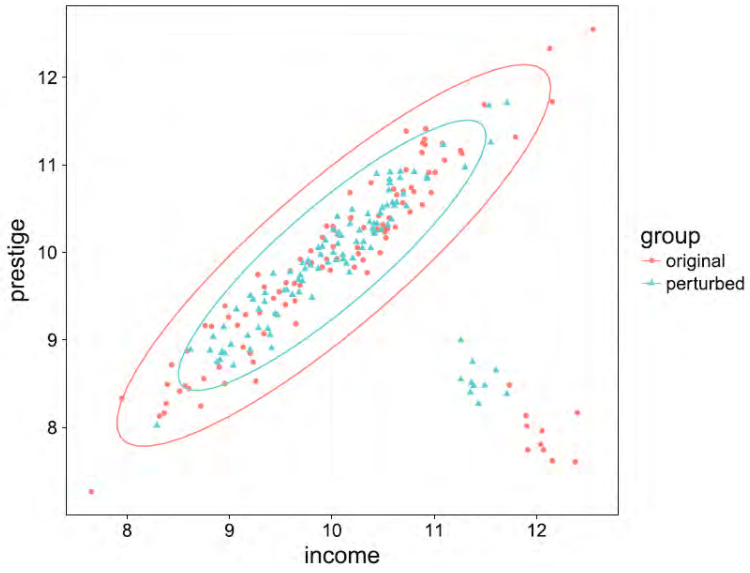


Abbildung 11: Kovarianzstruktur zu originalen Daten mit *correlated2* Noise Perturbation, ohne die Normalverteilungsannahme (aus Templ (2017, S. 128) entnommen)

3.2.2.4 *Outdect* Methode

Eine weitere Methode wurde ausschließlich für die Anwendung auf Outliers entwickelt, nämlich die als *outdect* implementierte Methode. Dies kommt zur Anwendung, wenn Outliers besser geschützt werden sollen bzw. müssen, um deren Offenlegungsrisiko zu reduzieren,"because outliers show a higher risk for reidentification than non-outliers"(Templ and Meindl, 2008, S. 115; vgl. Templ, 2017, S. 130).

Die Outliers werden basierend auf robusten Mahalanobis Distanzen mittels MCD-Schätzers detektiert (vgl. Benschop and Welch, 2019; Templ and Meindl, 2008, S. 115-116), d.h. "where the values of a variable x are greater than robust measure of location (e.g. the median) plus a robust measure of scatter (usually the Median Absolute Deviation)"(Templ and Meindl, 2008, S. 116).

Ferner wird die *outdect* Noise Methode mit Seed 2022 und Menge an Noise gleich 50 auf die bekannten Var1 und Var2 angewendet, was in Tabelle 12 präsentiert wird, wobei nur Beobachtungen 5 und 6 perturbiert werden:

	Var1	outdectNoise1	Var2	outdectNoise2
1	0.5	0.50	20	20
2	1.0	1.0	4	4
3	1.2	1.2	5	5
4	0.3	0.3	27	27
5	3.0	3.38	53	46.38
6	0.1	-0.40	11	0.34

Tabelle 12: Beispiel für Methode *outdect* in Noise Addition mit metrischen Variablen

4 Grundlagen in Machine Learning

> *"You will find it difficult to describe your mother's face accurately enough for your friend to recognize her in a supermarket. But if you show him a few of her pictures, he will immediately see the tell-tale traits. As they say, a picture — an example — is worth a thousand words. [...] Unable to define complicated concepts with adequate accuracy, we will convey them to the machine by way of examples. Unable to implement advanced skills, we instruct the computer to acquire them by systematic experimentation. Add to all this the ability to draw conclusions from the analysis of raw data, and to communicate them to the human user, and you know what the essence of machine learning is."* (Kubat, 2021, S. 1)

Ursprünglich aus dem Traum, dass Maschinen lernfähig sein können, kommt Machine Learning, aus dem großen Gebiet der künstlichen Intelligenz, heutzutage in ganz alltäglichen Anwendungen vor. Eine Welt ohne dies ist kaum mehr, wenn überhaupt noch vorstellbar. Von dieser Bedeutung und Prominenz ausgehend, werden in diesem Kapitel komprimierte Grundlagen in diesem Bereich dargestellt, um u.a. auch Vorgehen und Anwendung der Algorithmen in Kapitel 5 nachvollziehbar zu machen. Unterkapitel 4.1 fasst das Fundament zusammen, während in 4.2 die in 5 verwendeten Algorithmen beschrieben werden.

4.1 Fundament

Machine Learning kann in drei Teile aufgesplittet werden: 1) Supervised Learning, bei dem neben den Kovariablen X auch eine Zielvariable Y (Target) basierend auf den Kovariablen vorhergesagt wird; 2) Unsupervised Learning, bei dem es, anders als beim Supervised Teil, keine Zielvariable gibt, sondern z.B. Clusters; 3) Reinforcement Learning, bei dem es u.a. um Game Playing geht. Die vorliegende Arbeit befasst sich ausschließlich mit Supervised Learning, weswegen auf die anderen beiden Teile nicht näher eingegangen wird.

Die Zielvariable kann binär, kategorial oder metrisch bzw. kontinuierlich sein. Bei der binären Zielvariable ist der Output 0 oder 1 bzw. -1 oder 1. Die kategoriale ist die erwei-

terte Version der binären Zielvariable, indem mehr als zwei Klassen mögliche Outcomes sein können. Die metrische Zielvariable nimmt kontinuierliche Werte an.

Das Ziel des Targets ist in den meisten Fällen, die "genaueste Prädiktion" zu erreichen, welche von vielen Faktoren abhängt. Es gibt bereits einen wachsenden Unterbereich der künstlichen Intelligenz, der sich mit Methoden, Tools und Frameworks für das Verstehen und Interpretieren der Vorhersage des Machine Learnings befasst, nämlich Explainable Artifical Intelligence (XAI). In diesem Kontext hat Machine Learning auch den statistischen Bezug, Zusammenhänge zwischen den Variablen und allgemeinen Strukturen zu erkennen, was für diese Arbeit ebenfalls von Interesse ist. Auf Details im Hinblick auf XAI wird hier dennoch nicht eingegangen.

Das Vorgehen: Der Datensatz wird in Trainings- und Testdaten aufgesplittet, was zufällig, mit einer üblichen Proportion von jeweils 80:20 und mit einem festzulegenden Seed erfolgt. Des Weiteren, im Falle einer unbalancierten Zielvariable, muss sie zusätzlich stratifiziert werden. Das Aufsplitten ist deshalb so wichtig, um die zu trainierenden Daten von den zu prädiktierenden zu trennen. Ferner können sogenannte Hyperparameter zum Einsatz kommen, welche manuell festzulegen sind und eine Art Konfiguration gestalten. Dies soll dabei helfen, die optimalsten Parameter eines Modells auszuwerten bzw. zu finden. Beispiele sind u.a. die Anzahl an Bäumen, die maximale Anzahl der Variablen und die Tiefe jedes Baums in einem Random Forest Algorithmus (siehe einige beschriebene Algorithmen in 4.2). Sind Hyperparameter festzulegen, werden sie mit den Trainingsdaten mittels Tuning (z.B. mit Kreuzvalidierung) optimiert, um die besten zu finden. Kreuzvalidierung ist eine Methode basierend auf z.B. dem Aufsplitten der Trainingsdaten in k kleinere Partitionen, sogenannte k-Folds (k-Fold Kreuzvalidierung); andere Versionen können aber auch herangezogen werden. Dabei bleiben die bereits separierten Testdaten dennoch unberührt. Bei der k-Fold Kreuzvalidierung wird das Modell auf den k-1 Folds als Trainingsdaten trainiert und auf der verbliebenen Partition evaluiert (als Testdaten noch im Trainingset gedacht). Nach diesem Schritt werden die Trainingsdaten mit den bereits optimalsten Hyperparametern gefittet, um die besten Parameter bzw. Gewichte zu erreichen. Abschließend werden die besten Parameter sowie Hyperparameter auf die unberührten Testdaten (Hold-Out) angewendet, um die Perfomance des finalen Modells zu schätzen.

Die Performance Metriken für die Evaluierung des finalen Modells hängen von der Art der Zielvariable ab. Bei metrischer Zielvariable können mögliche Metriken u.a. der Mean Squared Error (MSE) bzw. Root Mean Squared Error (RMSE) und Mean Absolute Error (MAE) sein, welche in Tabelle 13 genauer definiert sind. Dabei sind Y der Vektor der beobachteten Outcome Variable mit den y_i beobachteten Werten, $\hat{Y}$ der Vektor der prädiktierten Outcome Variable mit den $\hat{y}_i$ prädiktierten Werten, $\bar{y}$ der beobachtete

Mittelwert der Response und n die Anzahl der Beobachtungen.

Kriterium	**Maßnahme**	**Definition**	**Schätzung**
RMSE	Genauigkeit der Schätzungen	$\sqrt{\mathbb{E}[(\hat{Y} - Y)^2]}$	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - Y)^2}$
R-squared	Anpassungsgüte eines Modells	1 - $\frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$	1 - $\frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$
MAE	Genauigkeit der Schätzungen	$\mathbb{E}\,\lvert\hat{Y} - Y\rvert$	$\frac{1}{n}\sum_{i=1}^{n}\lvert\hat{y}_i - Y\rvert$
Bias	Abweichung vom wahren Wert	$\mathbb{E}(\hat{Y})$ - Y	$\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i) - Y$

Tabelle 13: Performance Metriken für eine metrische Zielvariable

Im Falle einer binären Zielvariable können Metriken basierend u.a. auf der Confusion Matrix (14) herangezogen werden:

Confusion Matrix		Prädiktierte Klasse	
		0	1
Wahre Klasse	0	TN	FP
	1	FN	TP

Tabelle 14: Confusion Matrix für eine binäre Zielvariable

wobei TN für True Negative, FP für False Positive, FN für False Negative und TP für True Positive stehen. Daraus können folgende Metriken abgeleitet werden:

- Accuracy = $\frac{(TN+TP)}{(TN+FP+FN+TP)}$
- Precision = PPV = $\frac{TP}{(FP+TP)}$
- Recall = Sensitivity = TPR = $\frac{TP}{(FN+TP)}$
- Specificity = TNR = $\frac{TN}{(TN+FP)}$
- F1 Score = 2 $\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$

Zudem können auch Receiver Operating Characteristic (ROC) Kurven und Akaike Information Criterion (AIC) betrachtet werden. Erstgenanntes ist eine Darstellung, bei der die horizontale Achse die False Positive Rate (1 - Specificity), während die y-Achse die Sensitivity repräsentiert. AIC ist ein weiteres verbreitetes Kriterium, welches einen Kompromiss zwischen Anzahl an Kovariablen und Maximum Likelihood bildet, da mit steigender Anzahl an Variablen auch die Maximum Likelihood steigt, unabhängig davon, ob die hinzugefügte Variable einen Effekt auf die Zielvariable hat oder nicht.

Ferner, noch im Zusammenhang mit der Erreichung der "genauesten Prädiktion" (wie bereits angesprochen), sind weitere Begriffe von Bedeutung, auf die ebenso Rücksicht genommen werden muss, nämlich Overfitting und Underfitting. Erstgenanntes trifft zu, wenn sich das Modell zu gut an die Trainingsdaten anpasst und das "Rauschen" bzw. irrelevante Informationen "lernt", was wiederum zu einer Überanpassung und keiner guten Generalisation auf neue Daten führt, wofür es eigentlich gedacht ist. Dies kann u.a. passieren, wenn das Modell z.B. zu lange auf Trainingsdaten trainiert oder wenn es zu komplex ist. Underfitting ist das Gegenteil davon. Beides soll möglichst vermieden werden, was z.B. auch in Bonaccorso (2018, S. 282) bestätigt wird: "it's preferable to achieve a slightly worse accuracy with a higher generalization ability". Somit soll ein Kompromiss bzw. die Balance zwischen Performance und Overfitting erreicht werden.

4.2 Algorithmen

Die in Kapitel 5 angewendeten ML-Algorithmen, nämlich Lineare Regression, K-Nächste-Nachbarn, Random Forest, Gradient Boosting Machine und Neuronale Netze, sind im Folgenden näher erläutert.

4.2.1 Lineares Modell (LM)

Das Lineare Modell ist ein sehr bekannter verbreiteter statistischer sowie ML-Algorithmus. Es versucht das Verhältnis zwischen Kovariablen X (Input) und Zielvariable Y (Output) mit der getroffenen Annahme zu verstehen, dass diese eine lineare Beziehung darstellen. Somit wird Y aus einer linearen Kombination der Kovariablen X berechnet. Im Falle von nur einer Variable X als Input wird es als einfaches Lineares Modell bezeichnet. Um ein multivariates Lineares Modell handelt es sich, wenn mehrere Variablen X als Input dienen. Diese Variablen sind in Spalten in einer sogenannten Design Matrix platziert, die auch die lineare Equation für die Vorhersage von Y gestalten. Die für dieses Ziel zu schätzenden p Koeffizienten sind mit dem griechischen Buchstaben β repräsentiert. Ein weiterer Koeffizient (β_0) kommt hinzu, welcher Intercept genannt wird. Das multivariate Lineare Modell kann folgendermaßen definiert werden:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \varepsilon$$

wobei für ε eine Normalverteilung mit $\mu_\varepsilon = 0$ und σ_ε angenommen wird.

In höheren Dimensionen (d.h. wenn mehr als eine Kovariable X vorhanden ist) wird die Linie, die den Zusammenhang kennzeichnet, als Ebene bzw. Hyperebene bezeichnet. Diese Linie wird basierend auf dem (Ordinary) Least Squares Error gesteuert.

4.2.2 K-Nächste-Nachbarn (KNN)

Basierend auf der instance-based learning Methodologie erfolgt die Inferenz bei diesem Algorithmus durch direkten Vergleich neuer mit bereits bestehenden Samples (die als Instanzen definiert sind) anstatt mit einem mathematischen Modell (vgl. Bonaccorso, 2018, S. 233).

KNN ist sowohl für Klassifikation und Regression als auch für Clustering geeignet. Der Algorithmus beruht auf einer Distanz-Funktion $d(x_1, x_2)$, die mit der Minkowski Distanz verallgemeinert werden kann (idem):

$$d_p(\bar{x}_1, \bar{x}_2) = (\sum_{j=1}^{n} |x_1^j - x_2^j|^p)^{1/p}$$

Häufig wird $p = 2$ genommen, was der klassischen euklidischen Distanz entspricht, während $p = 1$ die Manhattan Distanz bezeichnet. Daraus bestimmt der KNN-Algorithmus die k nächsten Nachbarn, d.h. die k mit der kleinsten Distanz zu jeder Beobachtung im trainierten Datensatz. Für die Klassifizierung bzw. Cluster neu dazugekommener Beobachtungen gibt es zwei Möglichkeiten. Sie basieren auf einem festgelegten k oder auf einem festgelegten Radius bzw. Threshold r, sodass die Nachbarschaft, deren Distanz kleiner als oder gleich der definierten ist, berechnet wird (vgl. Bonaccorso, 2018, S. 236).

Bonaccorso (2018, S. 237) stellt fest: "The philosophy of KNN is that similar samples can share their features". Einem neuen User können Produkte (wie z.B. Filme, Bücher, u.a.) der gleichen Kategorie mittels eines Empfehlungssystems vorgeschlagen werden, basierend auf dem meisten ähnlichen geclusterten User und dessen konsumierten Produkten (idem).

4.2.3 Random Forest (RF)

Ein Random Forest ist ein Ensemble bzw. eine Aggregation von vielen Decision Trees (Entscheidungsbäumen), beruhend auf dem Bagging-Prinzip. In einem Entscheidungsbaum wird die Trainingsstichprobe, basierend auf den unabhängigen Variablen, in homogene Gruppen aufgeteilt. Durch eine rekursive Auswahlmethode wird das optimalste Attribut solange herangezogen und zur Bildung der Leaf Nodes (Blattknoten) aufgeteilt, bis ein festzulegendes Stopp-Kriterium erreicht wird. Die finale Vorhersage wird durch die durchschnittliche Vorhersage aller Bäume berechnet.

Geeignet sowohl für kategoriale als auch für metrische Variablen, können bei Random Forest einige Hyperparameter durch Kreuzvalidierung festgelegt werden, wie z.B. die Anzahl an Bäumen, die maximale Anzahl an Variablen, die Größe der Endknoten und die

Tiefe jedes Baums.

Stehend für Bootstrap Aggregation und entwickelt von Breiman (2001) werden bei Bagging zunächst B festzulegende Teilmengen aus den ursprünglichen Trainingsdaten gebootstrappt (d.h. gesampelt mit Zurücklegen). Einige Elemente können dann mehrmals in der Teilmenge auftauchen, während andere überhaupt nicht vorkommen werden. Bei der Berechnung der Auswahlsplits wird nur ein zufälliger Teil der Variablen anstelle von allen berücksichtigt. Übliche Wahlmöglichkeiten diesbezüglich sind die gerundete Quadratwurzel, log2 oder der natürliche Logarithmus (Bonaccorso, 2018, S. 282). Dieser Ansatz "schwächt" jeden Learner (deshalb auch "weak Learner" genannt), ermöglicht jedoch gleichzeitig eine deutliche Reduzierung der Varianz und des Biases sowie eine Erhöhung der Genauigkeit (idem). Sind letztendlich alle Modelle trainiert worden, kann die finale Vorhersage als Durchschnitt ermittelt werden.

Darüber hinaus ist bei Random Forest auch die Variable Importance von großer Bedeutung. Sie kann z.B. mittels Impurity Reduktion oder permutationsbasiert erfolgen. Erstgenanntes "is a measure proportional to the impurity reduction that a particular feature allows us achieve"(Bonaccorso, 2018, S. 286), definiert wie folgt:

$$Importance(\bar{x}^{(j)}) = \frac{1}{B}\sum_{b=1}^{B}\sum_{g}\frac{n(g)}{p}\Delta I_g^j$$

wobei B die gebootstrappten Teilmengen, $n(g)$ die Anzahl der Samples, die den Knoten g erreichen, p die gesamte Anzahl der Variablen und ΔI_g^j die am Knoten g nach dem Split der Verwendung der Variable j erreichte Impurity Reduzierung bezeichnen.

Die zweite Variante der Berechnung der Variable Importance ist mittels Permutation der Kovariablen. Dabei wird für jeden Baum die Vorhersage-Genauigkeit (im Falle einer Klassifikation) bzw. der MSE (für eine Regression) anhand der Out-of-Bag Daten vor und nach dem Permutieren jeder Kovariable berechnet. Anschließend werden die entstandenen Differenzen zwischen beiden Genauigkeiten bzw. MSEs über alle Bäume gemittelt und mit dem Standardfehler normalisiert (siehe Kuhn, 2019, Kapitel 15). Diese Version ist in einigen `R` Packages implementiert und kommt in Kapitel 5 bei der Anwendung der ML-Ansätze sowie deren Ergebnisse in 6 zum Einsatz.

4.2.4 Gradient Boosting Machine (GBM)

Gradient Boosting Machine ist ähnlich zu Random Forest, der Unteschied liegt jedoch zum einen darin, dass die Samples unverändert zwischen den Bäumen bleiben, während nur die Gewichte der einzelnen Beobachtungen modifiziert werden. Zum anderen werden die finalen Ergebnisse unterschiedlich aggregiert bzw. berechnet: Während bei RF dies am

Ende des Prozesses geschieht, erfolgt es bei GBM entlang des Aufbaus. "Gradient boosting means combining weak and average predictors to acquire one strong predictor" (Sanz, 2019, S. 129), was zur Robustheit führt.

Beim Boosting-Teil des GBM-Algorithmus werden die Bäume solange nacheinander basierend auf den Informationen der vorherigen Bäume trainiert, bis ein Stopp-Kriterium erreicht ist (wie z.B. eine bestimmte Anzahl an Bäumen) (idem). Aufgrund der Sequenzialität des Aufbaus wird jeder Learner optimiert, um die Genauigkeit des vorherigen zu erhöhen, wie auch Bonaccorso (2018, S. 307-308) hervorhebt: "[...] the effect of an update is to reduce the value of the global loss function by forcing the next model to improve its accuracy with respect to its predecessors". Daraus ergibt sich auch der Gradient-Teil des GBMs, nämlich aus der Minimierung des Gradienten der Verlustfunktion, während der Algorithmus jeden Baum aufbaut.

4.2.5 Neuronale Netze (NN)

Neuronale Netze versuchen mittels künstlicher Neuronen das Prinzip des menschlichen Gehirns zu imitieren. Diese Neuronen sind wiederum Gleichungen, die Inputs aufnehmen, deren Werte mit einer Reihe von Gewichten multiplizieren und deren Output an das nächste Neuron weiterleiten. Dieser Prozess erfolgt solange, bis ein einzelner Output bzw. eine Reihe von Outputs erreicht wird.

Multi-Layer Perceptron (MLP) ist eine Art künstlicher neuronaler Netze, die aus vielen verschiedenen, miteinander verbundenen Perceptrons (d.h. mathematische Gleichungen) bestehen, die wiederum mit anderen Perceptrons mittels Gewichten verbunden sind. Ein MLP ist zum einen ein Feed-Forward-Neuronales Netzwerk, d.h. die bereitgestellten Inputinformationen bewegen sich nur in Vorwärtsrichtung. Zum anderen ist es eine Art Deep Learning Algorithmus, mit dem Unterschied, dass Letztgenannter mehr "complexity in the calculations and the number of hidden layers" (Sanz, 2019, S. 134) hat.

Ein MLP enthält üblicherweise drei Layers, gemäß der von Bonaccorso (2018, S. 328) entnommenen Abbildung 12: 1) Input Layer, nämlich die Variablen des Datensatzes (keine Berechnung findet hier statt); 2) Hidden Layer, welche Berechnungen durchführen und die Gewichte bzw. Informationen vom Input- auf den folgenden Layer übertragen; 3) Output Layer, die den Output des Netwerks berechnen basierend auf den Inputs des Hidden Layers.

Jeder Knoten ist ein Neuron, welches mit Ausnahme der Inputknoten des Input Layers eine nichtlineare Activation Funktion verwendet, deren Zweck darin liegt, "to transform the input signal into an output signal that models complex nonlinear patterns" (Sanz, 2019, S. 133). Zunächst wird bei einem Neuron die gewichtete Summe der

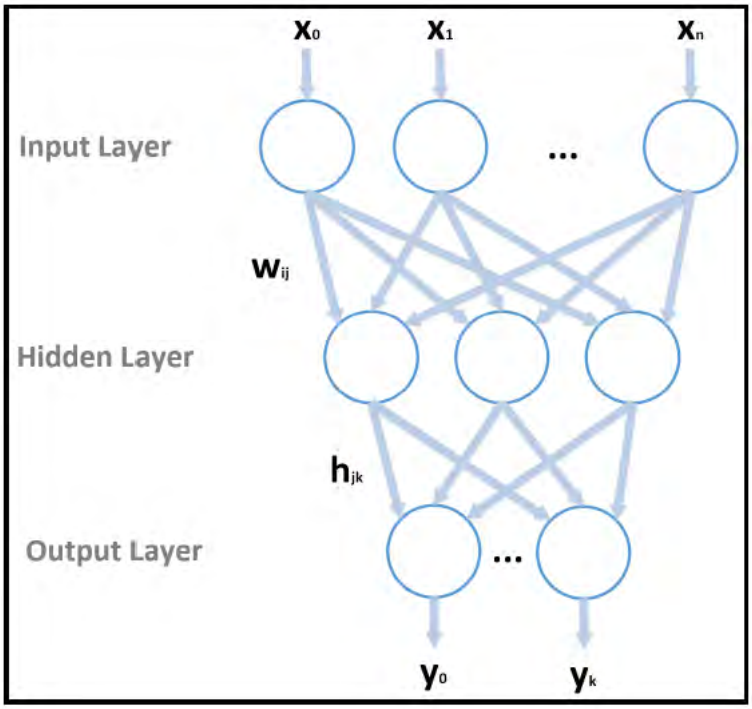


Abbildung 12: Darstellung des Multi-Layer Perceptrons (aus Bonaccorso (2018, S. 328) entnommen)

Inputs berechnet wie im Folgenden definiert ist. Dann kommt am Ende noch ein Bias hinzu, welcher ein Offset des finalen Ergebnisses der Gleichungen mit den Gewichten ist und manchmal gebraucht wird, um das richtige Ergebnis zu erhalten. Anschließend wird der skalare Wert in die Activation Funktion (wie z.B. Sigmoid-, ReLu- oder Tanh Funktion, u.a.) eingesetzt, welche den Output "vorbereitet".

In Abbildung 12 gibt es zwei Gewichtsmatrizen (W und H) und zwei entsprechende nicht illustrierte Bias Vektoren (b und c). Mit m Hidden Neuronen, f_h und f_a Activation Funktionen, $X_i \in \mathbb{R}^{n\times 1}$ und $Y_i \in \mathbb{R}^{k\times 1}$ kann der ganze Prozess folgendermaßen definiert werden (siehe z.B. Bonaccorso (2018), S. 329):

$$\begin{cases} z = f_h(W^T X + b), \text{ wobei } W \in \mathbb{R}^{n\times m} \text{ und } b \in \mathbb{R}^{m\times 1} \\ Y = f_a(H^T z + c), \text{ wobei } H \in \mathbb{R}^{m\times k} \text{ und } c \in \mathbb{R}^{k\times 1} \end{cases}$$

Die Gewichte werden anfangs zufällig gesetzt und während des Ablaufs mittels Berechnung ihrer partiellen Ableitungen korrigiert bzw. angepasst, um den Verlust zu minimieren (siehe z.B. Bonaccorso (2018, S. 322)). "These weights specify the number of errors that occur when processing the input, which is obtained by comparing the expected output"(Sanz, 2019, S. 134).

Der Trainingsprozess bei MLP erfolgt mittels einer Art von Gradient Descent, deren

Gradienten wiederum mittels Backpropagation berechnet werden. Während bei einem Klassifikationstask die Cross-Entropy Verlustfunktion minimiert wird und im multiplen Klassifikationsfall die Softmax als Output Funktion zur Anwendung kommt, gibt es keine Activation Funktion beim Output Layer in einer Regression.

5 Anwendung von ML-Algorithmen und Anonymisierungsverfahren

Dieses Kapitel wird in zwei Teile aufgesplittet, nämlich mit einer metrischen und einer binären Zielvariablen. Bei Erstgenanntem (in 5.1) werden die in Kapitel 4 dargestellten ML-Verfahren auf perturbierten Daten, welche mittels einiger in Kapitel 3 beschriebener Anonymisierungsverfahren generiert worden sind, durchgeführt, anhand ihrer Güte untersucht und verglichen. Als Performance Evaluation für die metrische Zielvariable werden der RMSE für alle Algorithmen und die Variable Importance (für RF und GBM) in Betracht gezogen. Für die binäre Zielvariable (siehe 5.2) wird eine Logistische Regression herangezogen, deren Beschreibung ebenso erläutert wird. Zudem wird PRAM (in 3.1.3 beschrieben) zum einen auf eine binäre Zielvariable und zum anderen auf eine binäre Kovariable angewendet. Anschließend wird auch der EM-Algorithmus erläutert, der in 6.2.2 zur Anwendung kommt. Das alles wird überwiegend auf mathematische Art untersucht.

5.1 Metrische Zielvariable

Dieses Unterkapitel befasst sich mit der Beschreibung der angewendeten Daten in einem praktischen Fall mit einer metrischen Zielvariable, der Anwendung von Anonymisierungsverfahren sowie von ML-Algorithmen und abschließend den Performance Metriken.

5.1.1 Daten

Der für die Öffentlichkeit zugängliche synthetische Datensatz *eusilc* aus dem Package `laeken` von Alfons and Templ (2013) mit metrischer Zielvariable hat sich für die Anwendung dieser Arbeit als geeignet erwiesen. Dieser wurde, der `R` Dokumentation zufolge, synthetischerweise vom echten ursprünglichen österreichischen EUSILC (European Union Statistics on Income and Living Conditions) Datensatz generiert. Die EUSILC Panelerhebung wird jährlich in den EU-Mitgliedstaaten und anderen europäischen Ländern durchgeführt. Der Datensatz enthält 14.827 Beobachtungen, davon 2.720 Beobachtungen mit fehlenden Werten (27.200 NAs insgesamt, was ungefähr 6,55% der Daten entspricht), und folgende 28 Variablen, wie in Tabelle 15 aufgelistet. Dabei ist py010n (Einkommen) die metrische Zielvariable für die spätere Anwendung von ML-Algorithmen.

Name	Variable	Ausprägung
db030	household ID	[1; 6.000]
hsize	Anzahl der Personen im Haushalt	[1; 9]
db040	Bundesland	Burgenland, Carinthia, Lower Austria, Salzburg, Styria, Tyrol, Upper Austria, Vienna und Vorarlberg
rb030	personal ID	[1; 600.002]
age	Alter der Person	[-1; 97]
rb090	Geschlecht	male und female
pl030	wirtschaftlicher Status	1 = working full time, 2 = working part time, 3 = unemployed, 4 = pupil, student, further training or unpaid work experience or in compulsory military or community service, 5 = in retirement or early retirement or has given up business, 6 = permanently disabled or/and unfit to work or other inactive person, 7 = fulfilling domestic tasks and care responsibilities
pb220a	Staatsangehörigkeit	AT, EU und Other
py010n	bares oder bargeldähnliches Einkommen der Mitarbeiter	[0; 151.894]
py050n	Geldleistungen oder Verluste aus selbständiger Erwerbstätigkeit	[-1.653; 112.073]
py090n	Arbeitslosengeld	[0; 27.354,3]
py100n	Altersvorsorge	[0; 75.837]
py110n	Hinterbliebenenversorgung	[0; 21.281,02]
py120n	Krankengeld	[0; 31.472,4]
py130n	Invaliditätsleistungen	[0; 52.480]
py140n	Ausbildungsbezogene Zulagen	[0; 19.440,37]
hy040n	Einkünfte aus der Vermietung einer Immobilie oder eines Grundstücks	[0; 118.083,6]
hy050n	Familien-/Kindergeld	[0; 96.449]
hy070n	Wohngeld	[0; 4.871,81]
hy080n	regelmäßig erhaltene Geldtransfers zw. Haushalten	[0; 107.951,9]
hy090n	Zinsen, Dividenden, Gewinne aus Kapitalanlagen	[0; 109.857,8]
hy110n	Einkommen der unter 16-Jährigen	[0; 13.627,23]
hy130n	regelmäßig gezahlte Geldtransfers zw. Haushalten	[0; 49.285,3]
hy145n	Einnahmen für Steuerausgleich	[-28.690,3; 27.018,1]
eqSS	äquivalente Haushaltsgröße	[1; 4,5]
eqIncome	leicht vereinfachte Version des Äquivalenzeinkommens der Haushalte	[0; 152.208]
db090	Haushaltsstichprobengewichte	[357.9; 1.032]
rb050	Persönliche Probengewichte	[357.9; 1.032]

Tabelle 15: EUSILC Datensatz

Für das weitere Vorgehen hinsichtlich des SDC-Prozesses wurden Variablen db030 und rb030 gelöscht, da sie direkte Identifikatoren sind (siehe Abschnitt 2.3 für die bereitgestellte Definition). Ferner wurden die Variablen rb050 und eqSS ausgeschlossen, deren Wichtigkeit für unbedeutend angenommen wurde. Zudem wurde ein Korrelationsplot (siehe Abbildung 13) mithilfe vom Package `ggplot2` von Wickham (2016) erstellt, um einen Überblick über die Korrelationsstruktur der Daten zu erhalten.

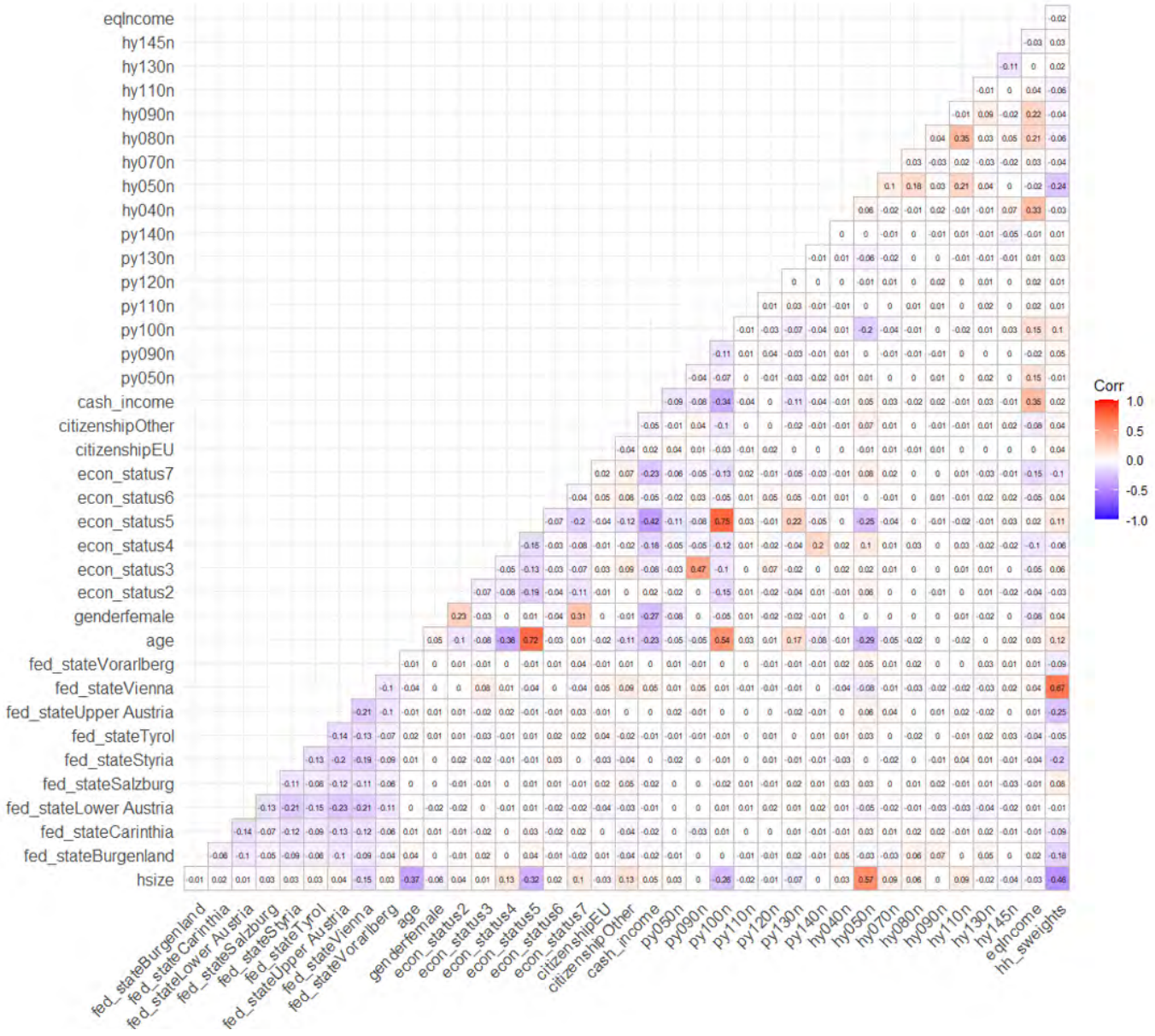


Abbildung 13: Korrelationsplot der EUSILC Daten

Bei der Variable eqIncome stellte sich zunächst die Frage, ob sie angemessen als Erklärungsvariable für die Vorhersage der Zielvariable py010n (bereits als cash_income umbenannt) wäre, da sie per se das Einkommen alleine erklären könnte und keine wei-

teren ML-Algorithmen benötigt würden. Basierend auf dem Korrelationsplot wurde sie trotzdem mit aufgenommen, da der Wert zwischen cash_income und eqIncome in Höhe von 0.36 eine schwache Korrelation aufweist.

5.1.2 Anwendung von Anonymisierungsverfahren

Für die Prozedur der Anonymisierung der Daten kommt das Package `sdcMicro` zur Anwendung. Zunächst wird ein SDC-Objekt mithilfe der Funktion *createSdcObj()* erstellt, indem als (kategoriale) Key Variablen fed_state, gender, hsize und econ_status festgelegt werden. Zudem wird mit den numerischen Variablen (Argument *numVars* in der gerade erwähnten Funktion) herum experimentiert. Aus der Kombination eqIncome, age, hy080n, py100n und py050n werden die Variablen für die Perturbation der Daten aufbauend der Reihe nach selektiert. Die Zielvariable cash_income wird nicht perturbiert. Des Weiteren ist auch die Festlegung der gewichteten Variable anzugeben (in dem Fall hh_sweights), da es sich um eine Haushaltsstruktur handelt.

Die fünf gerade genannten Variablen wurden deshalb gewählt, um verschiedene Korrelationsstrukturen sowie deren Auswirkungen auf die Performance der Algorithmen nach der Perturbation zu untersuchen. Basierend auf Abbildung 13 ist eqIncome schwach mit zwei Variablen korreliert, nämlich mit cash_income (corr = 0.35) und mit hy040n (corr = 0.33), und mit anderen kaum bzw. gar nicht. Age weist zwei mittelstarke Korrelationen auf, nämlich mit econ_status5 (corr = 0.72) und py100n (corr = 0.54), sowie drei weitere schwache Korrelationen mit anderen (-0.29, -0.36 und -0.37). Hy080n und py050n weisen keine Korrelation mit den Variablen auf. Die Korrelation zwischen py100n und age und zwischen py100n und econ_status5 beträgt jeweils 0.54 und 0.75.

Anschließend beginnt der Anonymisierungsprozess mit den in Kapitel 3 beschriebenen Verfahren. Als Erstes wird die kategoriale Variable hsize umkodiert, indem *TopBotCoding()* appliziert wird. Bei diesem Schritt werden die Kategorien 6 bis 9 kombiniert, da mittels eines Histrogramms (siehe 14) festgestellt wurde, dass diese sehr wenige Beobachtungen enthielten und dadurch höheres Offenlegungsrisiko aufwiesen.

Local Suppression für eine weitere Reduzierung des Risikos kommt in der gesamten Arbeit nicht zur Anwendung, da ML-Algorithmen sehr sensibel auf fehlende Werte reagieren. Das bedeutet, dass Beobachtungen mit den durch Local Suppression generierten fehlenden Werten anschließend ohnehin gelöscht werden müssten bzw. nicht benutzt werden könnten, was zu einem Informationsverlust führen würde.

PRAM wurde zwar für einen eventuell späteren Zeitpunkt reserviert (im Sinne von Überlegungen, dieses Verfahren ebenfalls noch einzusetzen), wurde letztendlich jedoch nicht verwendet.

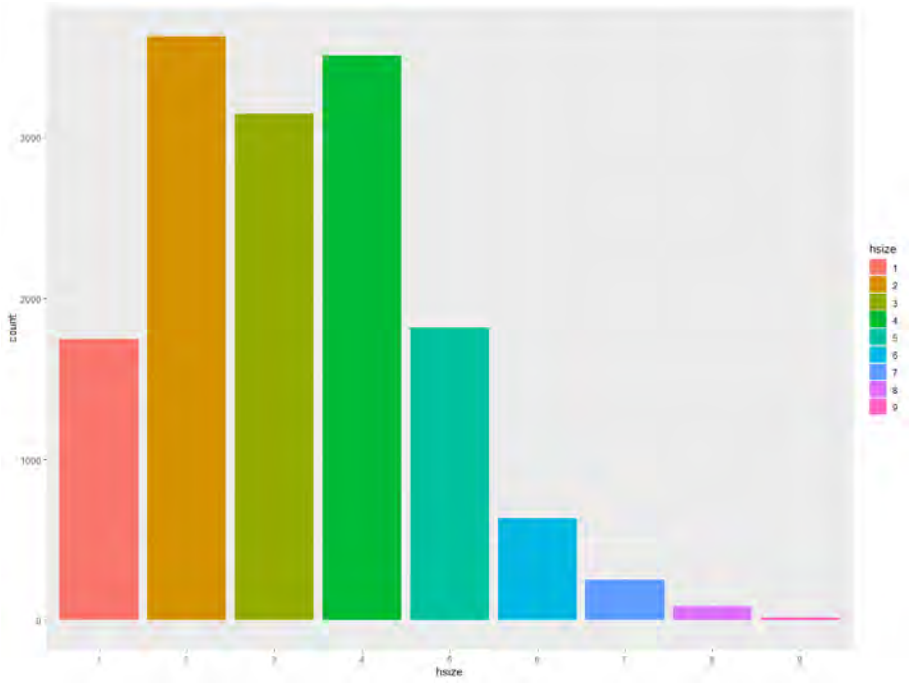


Abbildung 14: Verteilung der Variable hsize

Bei Microaggregation wurde zum einen das Aggregationslevel und zum anderen die Microaggregationsmethode variiert. Erstgenanntes wurde mit 3, 10 und 15, und bei Letztgenanntem die Methoden *mdav* und *simple* ausprobiert. Außerdem kommen noch die Kombinationen bzw. die Variationen der zu perturbierenden Kovariablen dazu, aufbauend wie im ersten Abschnitt angesprochen. D.h. erst wird Microaggregation nur auf eqIncome angewendet, dann auf eqIncome und age gleichzeitig, als Nächstes auf eqIncome, age und hy080n usw. bis alle 5 oben erwähnten Variablen angewendet worden sind.

Als weiteres Anonymisierungsverfahren ist Noise Addition zum Einsatz gekommen. Dabei wurden die Methoden *additive* und *correlated2* verwendet. Letzterem wurde gegenüber *correlated* Version deswegen der Vorzug gegeben, weil mittels eines Jarque-Bera Tests festgestellt wurde, dass die Variablen die Normalverteilungsannahme verletzten. Die Menge an additiver Noise wurde zwischen 50 und 250 variiert (nämlich, 50, 100, 150, 200 und 250). Was die Methode *correlated2* anbelangt, lag die Variation von α zwischen 0.1 (dem minimalen Wert, den α annehmen kann) und 1 (dem maximalen Wert), nämlich 0.1, 0.25, 0.5, 0.75 und 1. Das Schema zur Auswahl der zu perturbierenden Variablen erfolgte analog wie bei der Microaggregation.

Eine weitere Methode, die eventuell auch angewandet werden könnte ist *outdect* hinsichtlich der (möglicherweise existierenden) Outliers. Mittels Cooks Distanz (*cooks.distance()* in `R`) wurde untersucht, ob einflussreiche Fälle zu finden sind. Cook and Weisberg (1982) sprechen von einer Grenze von 1 was die Bestimmung der Existenz von Outliers angeht.

Sind also die Cook Distanzen unter 1, existieren keine einflussreichen Fälle. Aus der Abbildung 15 kann entnommen werden, dass alle Distanzen deutlich unter 1 liegen, was somit auf keine Ausreißer hinweist. Jedoch stechen 3 Beobachtungen hervor, nämlich 13071, 273 und 5630.

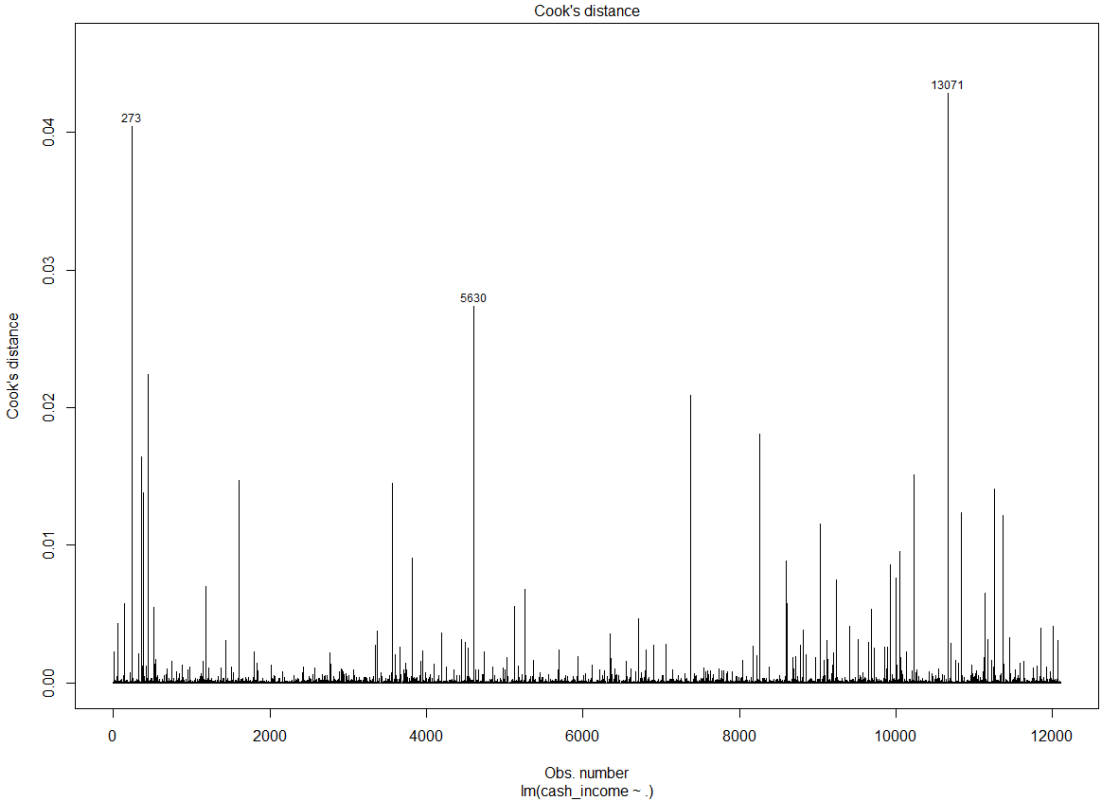


Abbildung 15: Cooks Distanzen der EUSILC Daten

5.1.3 Anwendung von ML-Algorithmen

Fünf ML-Algorithmen kamen zur Anwendung, nämlich Random Forest, Lineare Regression, Neuronale Netze, K-Nächste-Nachbarn und Gradient Boosting Machine, die auch in Kapitel 4.2 beschrieben worden sind. `caret` von Kuhn (2022) ist ein wesentliches verwendetes Package beim Trainieren und Testen der Algorithmen.

Zunächst wurden die ursprünglichen Daten ohne Perturbation trainiert, um die Performance vor und nach der Anwendung der Anonymisierungsverfahren vergleichen zu können. Dafür wurde der EUSILC Datensatz (schon ohne die 4 Variablen wie in 5.1.1 beschrieben) zuerst in Trainings- und Testdaten (mit Anteil 80:20 jeweils) aufgesplittet (siehe in 4.1 die beschriebene Idee des Vorgehens). Anschließend wurde die Formula für

das Training festgelegt, die wie folgt ist:

$$\begin{aligned} cash\_income \sim & hsize + fed\_state + age + gender + econ\_status + citizenship + \\ & py050n + py090n + py100n + py110n + py120n + py130n + \\ & py140n + hy040n + hy050n + hy070n + hy080n + hy090n + \\ & hy110n + hy130n + hy145n + eqIncome + hh\_sweights \end{aligned} \tag{6}$$

Danach wird jeder der eben genannten Algorithmen 100 Mal wiederholt und davon jeweils der Mittelwert sowie der Median berechnet. D.h. es werden 100 Random Forests, 100 Lineare Regressionen usw. trainiert. Für die Vorhersage auf Basis der Testdaten wird aus den 100 trainierten Modellen das Modell mit dem kleinsten RMSE ausgewählt.

Das Training und Testen der perturbierten Daten erfolgt analog. Der Unterschied bei der verwendeten Formula im Vergleich zu der in (6) liegt darin, dass nun eine oder mehrere der fünf Variablen (nämlich age, py050n, py100n, hy080n und eqIncome) perturbiert sind. Zudem wurde eine Funktion geschrieben, in welcher der ganze Prozess aufgebaut wird - basierend auf dem anzugebenden Input (algorithm, repetitions, variables_addNoise bzw. variables_microaggr, microaggr_k, microaggr_method, addNoise_noise, addNoise_method und delta (für Methode *correlated2*)) - und die schlussendlich die Performance Metriken ausgibt.

5.1.4 Performance Metriken

Die Performance Metriken evaluieren die verwendeten Algorithmen bzw. deren Sensibilität auf die anonymisierten Daten. Für die Trainings- und Testdaten werden die Metriken Root Mean Squared Error (RMSE), R-squared und Mean Absolute Error (MAE) berechnet (deren Definitionen sind in Tabelle 13). Für die Testdaten wird noch ein Bias zusätzlich geschätzt.

Darüber hinaus wird bei Random Forest und Gradient Boosting Machine die Variable Importance mittels *varImp()* aus dem Package `caret` berechnet. Die Importance Scores werden automatisch auf einen Wert zwischen 0 und 100 skaliert. Dies wird für Random Forest gemäß Dokumentation des Packages folgendermaßen berechnet: "the MSE is computed on the out-of-bag data for each tree, and then the same computed after permuting a variable. The differences are averaged and normalized by the standard error"(Kuhn, 2022, Kapitel 15). Bei Gradient Boosting Machine erfolgt die Berechnung analog wie bei einem einzelnen Baum, die Werte werden nur für jede Boosting-Iteration aufsummiert.

5.2 Binäre Zielvariable

Dieser Abschnitt ist etwas anders als der vorherige aufgebaut. Statt einen praktischen Fall (im Sinne eines Datensatzes) zu betrachten, erfolgt hier eine eher mathematische Herangehensweise. Zudem werden Messfehler, die durch die Anwendung von PRAM verursacht worden sind, sowohl in einer binären Kovariable X (in 5.2.3) als auch in der Outcome Variable Y (in 5.2.2) erfasst. Da nun von einer binären Zielvariable ausgegangen wird, wird eine Logistische Regression als ML-Algorithmus in Betracht gezogen, deren Beschreibung nicht in 4.2, sondern in 5.2.1 Platz findet. Ferner wird der EM-Algorithmus in 5.2.4 erläutert, mit dessen Hilfe ein Korrekturansatz in 6.2.2 präsentiert wird.

5.2.1 Logistische Regression und Newton-Raphson-Algorithmus

Bei einer binären Zielvariable (d.h. $Y \in \{0, 1\}$) wird sehr häufig eine Logistische Regression herangezogen, um den Zusammenhang zwischen Outcome und Kovariablen zu fitten. Diese ist wie folgt definiert:

$$P(Y = 1|X) = \frac{\exp(\beta^t X)}{1 + \exp(\beta^t X)} = \frac{1}{1 + \exp(-\beta^t X)} \tag{7}$$

wobei Y die Zielvariable, X die Kovariablen, β^t den transponierten Vektor mit den zu schätzenden $(\beta_0, \beta_1, ..., \beta_p)$ Regressionskoeffizienten (mit p die Anzahl der Variablen) und $\beta^t X$ den linearen Prädiktor bezeichnen. Da Y und X nicht in der gleichen Skalierung sind, wird der lineare Prädiktor mittels einer sogenannten Responsefunktion so transformiert, dass das Ergebnis dieser Funktion wieder in der selben Skalierung von Y liegt und als Wahrscheinlichkeit, eine 1 zu beobachten, gesehen werden kann.

Die Outcome Variable Y folgt einer Binomial- bzw. Bernoulli-Verteilung, mit $\pi = P(Y = 1|X)$, d.h. $Y_i \sim B(1, \pi_i)$, was dazu führt, dass die Likelihood Funktion folgendermaßen bestimmt ist:

$$\mathcal{L}(\pi) = \prod_{i=1} f(y_i|\pi) = \prod_{i=1} \pi^{y_i}(1 - \pi)^{1-y_i} \tag{8}$$

Für jeden Datenpunkt y_i wird ein bestimmtes π_i anstatt π verwendet, welches von Beobachtung $x_i = (x_{i0}, x_{i1}, ..., x_{ip})$ abhängt, was zu $\pi_i = h(\beta^t x_i)$ führt, mit h einer Responsefunktion $h : \mathbb{R}^n \to [0, 1]$. Somit kann die Likelihood Funktion (8) wie folgt definiert werden:

$$\mathcal{L}(\beta) = \prod_{i=1} f(y_i|x_i, \beta) = \prod_{i=1} \pi_i^{y_i}(1 - \pi_i)^{1-y_i} \tag{9}$$

Der Wert, der die Likelihood Funktion (9) maximiert, ist der Maximum Likelihood

Schätzer für β. Sehr häufig wird die log Likelihood Funktion maximiert:

$$l(\beta) = \log \mathcal{L}(\beta) = \sum_{i=1}^{n} \Big(y_i \log(\pi_i) + (1 - y_i) \log(1 - \pi_i) \Big) \tag{10}$$

Um das Maximum aus (10) zu erhalten, wird $\frac{\delta}{\delta \beta_j} l(\beta) = 0$ gesetzt (mit $j \in (0, 1, ..., p)$), was dennoch keine analytische Lösung hat, da die Gleichung nicht linear in den βs ist. Daher kann der Newton-Raphson-Algorithmus herangezogen werden (van den Hout, 1999, S. 61). Die Ableitung der Equation resultiert in die Score Funktion:

$$S(\beta) = \left(\frac{\delta}{\delta \beta_0} l(\beta), \frac{\delta}{\delta \beta_1} l(\beta), ..., \frac{\delta}{\delta \beta_p} l(\beta) \right) \tag{11}$$

Die Fisher Information $I(\beta)$, deren Inverse asymptotische Schätzungen der Kovarianz liefert, ist durch die negative Ableitung der Score Funktion zu erhalten:

$$I(\beta) = -\frac{\delta^2}{\delta \beta_j \delta \beta_s} l(\beta) = -\frac{\delta}{\delta \beta_s} S(\beta) \tag{12}$$

Initiierend mit einem Wert $\beta^{(0)}$ wird der Newton-Raphson-Algorithmus mittels der beiden letztgenannten Funktionen (11) und (12) solange iteriert, bis Konvergenz hinsichtlich β erreicht ist (idem):

$$\beta^{(1)} = \beta^{(0)} + (I(\beta^{(0)})^{-1} S(\beta^{(0)})$$

Darüber hinaus behauptet van den Hout (1999, S. 62) noch Folgendes: "when the likelihood function is adjusted correctly to account for the perturbation, logistic regression can proceed in the usual way, as though no perturbation has occurred". Diese Aussage ist für das Weitere von großer Relevanz, wenn PRAM als Anonymisierungsverfahren appliziert wird und besonders für die dargestellten Korrekturansätze.

5.2.2 Perturbation einer binären Zielvariable

Hier wird eine Situation betrachtet, in der eine binäre Zielvariable Y mittels PRAM (in 3.1.3 beschrieben) mit folgender Markov Matrix perturbiert wird:

$$\boldsymbol{P}_Y = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix}$$

Weiterhin gilt $\pi = P(Y = 1|X)$ wie in 5.2.1 beschrieben. Da die Zielvariable dennoch mittels PRAM basierend auf der Markov Matrix kontaminiert wird ($Y \rightarrow Y^*$),

werden manche Beobachtungen mit $Y^* = 1$, welche eigentlich ursprünglich $Y = 0$ waren, generiert, wie folgende Abbildung 16 veranschaulicht:

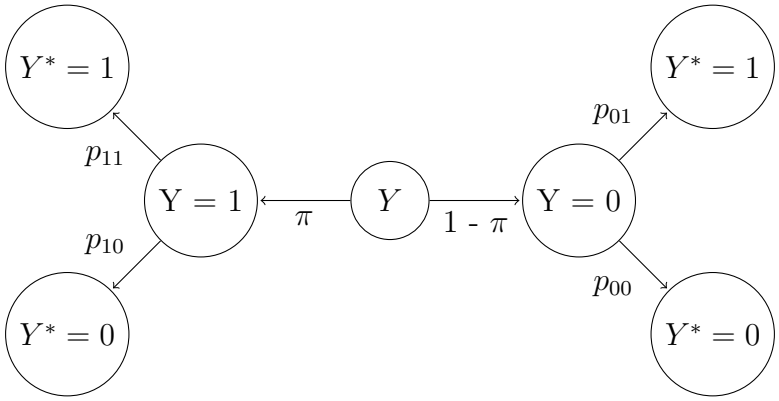


Abbildung 16: Perturbation einer binären Zielvariable

Nun wird die Wahrscheinlichkeit, eine 1 bei den perturbierten Daten zu beobachten, somit wie folgt berechnet:

$$P(Y_i^* = 1|X, \boldsymbol{P}_Y) = p_{11}P(Y_i = 1|X) + p_{01}P(Y_i = 0|X) = p_{01} + (p_{11} - p_{01})\pi_i \tag{13}$$

sowie

$$P(Y_i^* = 0|X, \boldsymbol{P}_Y) = p_{10}P(Y_i = 1|X) + p_{00}P(Y_i = 0|X) = p_{00} + (p_{10} - p_{00})\pi_i \tag{14}$$

Aus (13) und (14) erfolgt die (perturbierte) Likelihood Funktion:

$$\mathcal{L}^*(\beta) = \prod_{i=1} f(y_i^*|x_i, \beta, \boldsymbol{P}_Y) = \prod_{i=1}\Big(p_{01} + (p_{11} - p_{01})\pi_i\Big)^{y_i^*}\Big(p_{00} + (p_{10} - p_{00})\pi_i\Big)^{1-y_i^*} \tag{15}$$

wobei y_i^* ein perturbierter Datenpunkt, $x_i = (x_{i0}, x_{i1}, ..., x_{ip})$ und $\pi_i = h(\beta^t x_i)$ sind, wie vorher in 5.2.1 bestimmt. Anschließend wird die (perturbierte) log Likelihood Funktion berechnet:

$$l^*(\beta) = \log \mathcal{L}^*(\beta) = \sum_{i=1}^{n}\Big(y_i^*\log\Big(p_{01}+(p_{11}-p_{01})\pi_i\Big)+(1-y_i^*)\log\Big(p_{00}+(p_{10}-p_{00})\pi_i\Big)\Big) \tag{16}$$

Dieser Abschnitt schließt mit dem Link zu einem Korrekturansatz in diesem Kontext ab, welcher in 6.2.1 deskribiert ist.

5.2.3 Perturbation einer binären Kovariable

In diesem Teil wird eine Situation angenommen, in der nun eine binäre Kovariable durch PRAM kontaminiert wird und die (unperturbierte) Zielvariable ebenfalls binär ist, um die

Modellierung anhand der Logistischen Regression durchzuführen. Analog zur Zielvariable hat auch die binäre Kovariable zwei Kategorien angenommen, 0 und 1. Des Weiteren führt die Anwendung von PRAM auf X zu X^* mittels folgender Transitionsmatrix (ähnlich zu $\boldsymbol{P}_Y$ in 5.2.2):

$$\boldsymbol{P}_X = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix}$$

Dabei lautet die (perturbierte) Likelihood Funktion:

$$\mathcal{L}^*(\beta) = \prod_{i=1}^{n} f(y_i|x_i^*, \beta, \boldsymbol{P}_X)$$

Auch hier wird auf einen Korrekturansatz verwiesen, der in 6.2.2 zu finden ist.

5.2.4 EM-Algorithmus

Mit einem iterativen Vorgehen bei den Maximum Likelihood Schätzungen stellt der Expectation-Maximization (EM) Algorithmus eine Alternative zum Newton-Raphson-Algorithmus dar. Der EM-Algorithmus kommt häufig in statistischen Situationen mit unvollständigen Daten bzw. Daten mit Messfehlern zur Anwendung (siehe z.B. Yi, 2017, S. 58; van den Hout, 1999, S. 37). Ferner kommt er in 6.2.2 als ein Korrekturansatz einer mittels PRAM perturbierten binären Kovariable zum Einsatz, weswegen seine Beschreibung nun hier ihren Platz findet. van den Hout (1999, S. 64-66) präsentiert auch einen Korrekturansatz mittels seiner Anwendung bei einer durch PRAM kontaminierten Zielvariable.

van den Hout (1999, S. 38) beschreibt u.a. die allgemeine Form des EM-Algorithmus bei unvollständigen Daten. Yi (2017, S. 56-58) deskribiert ihn mit einem ähnlichen Schema in Bezug auf Daten mit Messfehlern. Auf Letztgenanntem wird basiert, um hier ebenfalls eine Beschreibung zu liefern. Des Weiteren seien θ der Vektor der zu schätzenden Parameter in einem Modell und die mit X bezeichneten beobachteten Kovariablen in zwei Komponenten aufgeteilt: X^* (fehlerhafter Teil) und Z (fehlerfrei). Basierend auf Y, X^* und Z kann es in etlichen Situationen etwas kompliziert sein, die beobachtete Likelihood ($\mathcal{L}_o(\theta)$)

$$\mathcal{L}_o(\theta) = \prod_{i=1}^{n} f(y_i|x_i^*, z_i) \tag{17}$$

zu schätzen, während die vollständige Likelihood ($\mathcal{L}_c(\theta)$) bezüglich Y, X, X^* und Z

$$\mathcal{L}_c(\theta) = \prod_{i=1}^{n} f(Y_i, X_i, X_i^*|Z_i) \tag{18}$$

etwas einfacher erfolgt. In diesen Fällen kommt der EM-Algorithmus zum Einsatz, dessen Prozedur eigentlich darin besteht, zwischen zwei Schritten zu iterieren, nämlich dem sogenannten Expectation Step (E-Step) sowie dem Maximization Step (M-Step).

Beim E-Step wird der bedingte Erwartungswert der log Likelihood (d.h. $Q(\theta, \theta^{(q)})$) geschätzt:

$$Q(\theta, \theta^{(q)}) = \mathbb{E}_{X|(Y,X^*,Z,\theta^{(q)})} \sum_{i=1}^{n} \log f(Y_i, X_i, X_i^*|Z_i, \theta^{(q)}) \tag{19}$$

Anschließend wird beim M-Step ein $\theta^{(q+1)}$ ausgewählt, sodass dieser Wert $Q(\theta, \theta^{(q)})$ in (19) maximiert. Die Prozedur zwischen E- und M-Step wird solange iteriert, bis Konvergenz hinsichtlich θ in der Iteration $\theta^{(q)}$ erreicht wird.

6 Ergebnisse

Dieses Kapitel befasst sich mit den Ergebnissen der sowohl metrischen als auch binären Zielvariablen aus dem vorherigen Kapitel 5. In 6.1.1 werden die Schlussfolgerungen des praktischen Falls mit den EUSILC Daten und in 6.1.3 die entsprechende Variable Importance erfasst. Darüber hinaus werden auch Simulationen durchgeführt, um die im praktischen Fall betrachteten Ergebnisse bei der metrischen Zielvariable zu bestätigen (siehe 6.1.2). Ferner erfolgt eine dritte Simulation in 6.1.4, die sich mit dem Offenlegungsrisiko und Informationsverlust befasst. Unterkapitel 6.1.5 befasst sich mit der SIMEX Methode als Korrekturverfahren für β bei einer metrischen kontaminierten Kovariable, während in 6.1.6 ein Korrekturansatz mittels Regressionskalibrierung für die individuellen Prädiktionen y_i im selben Kontext diskutiert wird. Des Weiteren werden auch Korrekturansätze hinsichtlich βs für eine binäre perturbierte Zielvariable und binäre Kovariable präsentiert (in 6.2.1 und 6.2.2). Zudem wurden zwei Korrekturmethoden hinsichtlich Adjustierung individueller Prädiktionen y_i ausprobiert (siehe 6.2.3 und 6.2.4).

6.1 Metrische Zielvariable

6.1.1 Fall betrachtete Ergebnisse hinsichtlich RMSE

Link zu allen Ergebnissen aller evaluierten Modelle auf den EUSILC Daten:
`https://docs.google.com/spreadsheets/d/1LzL-DLATzTn91aYtmVMON9UpgB80TSsPTe1LmAVjTgA/edit?usp=sharing`

Einiges konnte aus den EUSILC Daten festgestellt werden:

1. Variable Importance vs. Korrelation vs. RMSE:
   (a) Weist die Variable schwache bzw. keine Korrelation mit anderen Variablen auf und ist sie gleichzeitig sehr wichtig im Sinne von Variable Importance, dann steigt der RMSE deutlich. Das kann z.B. bei eqIncome und py050n festgestellt werden, welche einen Importance Score von jeweils 100 bzw. circa 31 besitzen.
   (b) Ist sie jedoch nicht sehr wichtig und gleichzeitg kaum mit anderen korreliert, dann gibt es kaum einen Unterschied beim RMSE, wie es z.B. bei Variable hy080n der Fall ist, mit einem Importance Score in Höhe von circa 3.

(c) Im Falle von hoher Variable Importance und zugleich mittelstarker bzw. starker Korrelation ist die Steigung des RMSEs eher geringfügig, wie beim Hinzufügen der Variablen age und py100n zu sehen ist, deren Variable Importance jeweils circa 27 bzw. circa 25 beträgt.

2. Ferner ist auch aufgefallen, dass sich bei Random Forest der RMSE der mit (noise = 250) additiver Noise Addition perturbierten Daten ungefähr dem RMSE der mit (delta (= α) = 1) *correlated2* Noise Addition annähert.
3. Darüber hinaus ist der RMSE bei Neuronalen Netzen unverändert geblieben, unabhängig davon ob originale Daten oder (mit allen ausprobierten Anonymisierungsmethoden) perturbierte Daten vorliegen.
4. Bei all den variierten Aggregierungsleveln in Microaggregation gab es keinen großen Unterschied im RMSE zu sehen.

6.1.2 Allgemeine Schlussfolgerungen der Ergebnisse hinsichtlich RMSE

Um die obigen Ergebnisse, insbesondere die Punkte 1 und 3, zu verallgemeinern, wurden hierzu zwei Simulationen aufgebaut.

Erste Simulation

Für die erste Simulation wurden fünf verschiedene Variablen (Y - Zielvariable - und vier Kovariablen) aus verschiedenen Verteilungen stammend mit jeweils 10.000 Beobachtungen generiert. Genauer dargestellt:

$$\begin{cases} var1 \sim N(\mu = 100,\ \sigma^2 = 80^2), \\ age \sim Pois(\lambda = 50), \\ var2 \sim \{U(min = 1, max = 30) + \gamma_{var1} \cdot 0,007 \cdot var1\}, \\ invests \sim \{[Par(n = 7.500, x = 10.000, \alpha = 4) + (n = 2.500, x = 0)] + \gamma_{age} \cdot 100 \cdot age\}, \\ Y \sim \{[Par(n = 7.000, x = 3.000, \alpha = 4) + (n = 3.000, x = 0)] + \beta_{var1} \cdot var1 + 10 \cdot age + \\ \qquad \beta_{var2} \cdot 5 \cdot var2 + 0,04 \cdot invests\}, \end{cases} \tag{20}$$

wobei die Parameter γ_{var1}, γ_{age}, β_{var1} und β_{var2} variiert werden, um das Verhältnis zwischen Variable Importance und Korrelation untersuchen zu können.

Anschließend erfolgte der (80:20) Split in Trainings- und Testdaten, analog zu 5.1.3. Dazu wurde ein 100-facher GBM-Algorithmus auf die einzelnen originalen Kovariablen angewendet und daraus der Mittelwert und Median berechnet. Diese Prozedur wurde zudem 100 Mal wiederholt, mit 100 verschiedenen festgelegten Seeds (082022-082121),

sodass 100 neue Datensätze erzeugt wurden. Dazu werden die Parameter γ_{var1}, γ_{age}, β_{var1} und β_{var2} variiert, deren Variationen und Ergebnisse in Tabelle 16 dargestellt sind. Ferner erfolgte die Perturbation der einzelnen Variablen mit additiver Noise (Noise = 250) und das Training und Testen wurde analog zu den originalen Daten durchgeführt. Darüber hinaus wurde auch auf die Variable Importance geachtet, sodass ein Threshold von 10 bestimmt wurde, um zu entscheiden, ob die Variable als wichtig angesehen werden kann oder nicht.

Die Ergebnisse hinsichtlich Variable Importance versus RMSE sowie die oben beschriebenen variierten Parameter werden in der folgenden Tabelle 16 zusammengefasst. Wie aus (20) ablesbar etabliert Parameter γ_{var1} die Korrelation zwischen var1 und var2. Dabei zeigt sich: Je höher dieser Wert, desto höher ist die Korrelation zwischen beiden; wird er auf 0 gesetzt, sind die zwei Variablen unkorreliert. Zudem baut Parameter β_{var2} die Variable Importance hinsichtlich var2 auf, sodass mit dessen steigendem Wert auch die Variable Importance von var2 steigt. Wird er auf 0 gesetzt, ist der Importance Score dieser Variable ebenso 0 (d.h. sie wird unwichtig). Das Analogon gilt für γ_{age} (Parameter, der die Korrelation zwischen age und invests reguliert) sowie für β_{var1} (Parameter, der die Variable Importance von var1 steuert).

Aus Tabelle 16 lässt sich Folgendes feststellen:

1. Ist die Variable unwichtig im Sinne von Variable Importance, unabhängig ob korreliert oder unkorreliert, gibt es kaum einen Unterschied beim RMSE nach der Perturbation. Dies ist in 1. und 6. zu sehen, indem Parameter $\gamma_{var1} = 1$ und $\gamma_{var1} = 10$ jeweils eine minimale bzw. starke Korrelation anzeigen, während $\beta_{var2} = 1$ auf einen kleinen Importance Score hindeutet.

2. Ferner, je wichtiger eine unkorrelierte zu perturbierende Variable wird, desto deutlicher steigt der RMSE. Das ist bei den Fällen 1. bis 3. ersichtlich, indem Parameter β_{var2} (der die Variable Importance steuert) steigt, während die Variable var2 mit Parameter $\gamma_{var1} = 1$ minimale Korrelation aufweist.

3. Ebenso höher wird der RMSE wenn die Variable Importance steigt, gesteuert durch den Parameter β_{var2} (1 = nicht wichtig, 10 = sehr wichtig), während die Variable gleichzeitig korreliert ist (mit Parameter $\gamma_{var1} = 10$). Das ist bei 4. und 6. ersichtlich, indem der RMSE sogar etwas höher als im unkorrelierten Fall (= 3.) ist.

4. Zudem, je stärker eine Variable korreliert wird ($\gamma_{var1} = \text{i}$, mit $\text{i} \in \{0 = \text{nicht korreliert}, 1 = \text{sehr schwach korreliert}, 10 = \text{starke Korrelation}\}$, die gleichzeitig sehr wichtig ist ($\beta_{var2} = 10$), desto höher wird der RMSE. Das ist bei den Fällen 3. bis 5. zu sehen.

additive perturb. var2	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1.	$\gamma_{var1} = 1$, $\beta_{var2} = 1$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2155,72	2155,51	4	0	0	96
Test	2160,51	2160,33	4	0	6	90
2.	$\gamma_{var1} = 1$, $\beta_{var2} = 5$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2156,38	2159,84	90	9	0	1
Test	2160,88	2164,76	55	44	0	1
3.	$\gamma_{var1} = 1$, $\beta_{var2} = 10$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2156,74	2165,17	2	98	0	0
Test	2161,46	2169,49	28	72	0	0
4.	$\gamma_{var1} = 10$, $\beta_{var2} = 10$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2157,09	2167,89	1	99	0	0
Test	2162,07	2172,22	16	84	0	0
5.	$\gamma_{var1} = 0$, $\beta_{var2} = 10$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2156,68	2164,96	3	97	0	0
Test	2161,54	2169,32	31	69	0	0
6.	$\gamma_{var1} = 10$, $\beta_{var2} = 1$, $\gamma_{age} = 1$, $\beta_{age} = 1$ und $\beta_{var1} = 1$					
Train	2155,77	2155,62	12	0	0	88
Test	2160,32	2160,44	10	2	4	84

Tabelle 16: Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable var2 über die 100 Seeds; Parameter γ_{var1} steuert die Korrelation während β_{var2} die Variable Importance bezüglich var2 kontrolliert

Situation 4 ist auch bei der Variable invests in Tabelle 17 zu sehen. Die Variable ist so konstruiert worden, dass sie immer sehr wichtig ist. Somit kann hier die Aussage erneut bestätigt werden, dass je korrelierter eine sehr wichtige Variable wird, desto höherer RMSE (siehe 1. bis 3., indem γ_{age} die Korrelation steuert).

additive perturb. invests	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1.	$\gamma_{age} = 0$, $\gamma_{var1} = 1$, $\beta_{age} = 1$, $\beta_{var2} = 1$ und $\beta_{var1} = 1$					
Train	2155,63	2166,94	0	100	0	0
Test	2160,48	2170,51	19	81	0	0
2.	$\gamma_{age} = 1$, $\gamma_{var1} = 1$, $\beta_{age} = 1$, $\beta_{var2} = 1$ und $\beta_{var1} = 1$					
Train	2155,72	2167,16	1	99	0	0
Test	2160,51	2170,93	18	82	0	0
3.	$\gamma_{age} = 10$, $\gamma_{var1} = 1$, $\beta_{age} = 1$, $\beta_{var2} = 1$ und $\beta_{var1} = 1$					
Train	2160,45	2181,9	0	100	0	0
Test	2164,99	2184,47	0	100	0	0

Tabelle 17: Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable invests über die 100 Seeds; Parameter γ_{age} steuert die Korrelation

Analog verhält es sich bei der Variable age in Tabelle 18. Situation 1 ist auch hier bei 1. zu sehen, nämlich kaum Veränderung beim RMSE wenn die Variable unwichtig ist. Situation 2 ist auch mit 1. und 4. dargestellt, da je wichtiger die unkorrelierte Variable age wird, umso mehr steigt der RMSE. Situation 3 lässt sich bei 2. und 3. ablesen, indem die sehr stark korrelierte Variable age wichtiger wird und somit auch der RMSE steigt.

additive perturb. age	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1.	$\gamma_{age} = 1$, $\beta_{age} = 1$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2155,72	2155,6	0	0	0	100
Test	2160,51	2160,57	0	0	2	98
2.	$\gamma_{age} = 10$, $\beta_{age} = 1$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2160,45	2164,87	1	1	30	68
Test	2164,99	2170,1	1	1	48	50
3.	$\gamma_{age} = 10$, $\beta_{age} = 20$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2163,73	2198,45	0	100	0	0
Test	2168,41	2201,47	0	100	0	0
4.	$\gamma_{age} = 1$, $\beta_{age} = 20$, $\gamma_{age} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2161,71	2198,28	0	100	0	0
Test	2166,01	2201,52	0	100	0	0

Tabelle 18: Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable age über die 100 Seeds; Parameter γ_{age} steuert die Korrelation während β_{age} die Variable Importance bezüglich age kontrolliert

Alle bisher gezogenen Schlussfolgerungen sehen bei var1 jedoch ganz anders aus. Hier gibt es bei keinem Fall eine bedeutende Veränderung des RMSEs. Warum?

additive perturb. var1	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1.	$\gamma_{var1} = 1$, $\beta_{var1} = 1$, $\beta_{var2} = 1$, $\beta_{age} = 1$ und $\gamma_{age} = 1$					
Train	2155,72	2155,73	11	0	0	89
Test	2160,51	2160,62	11	0	0	89
2.	$\gamma_{var1} = 1$, $\beta_{var1} = 10$, $\beta_{var2} = 1$, $\beta_{age} = 1$ und $\gamma_{age} = 1$					
Train	2157,94	2157,95	100	0	0	0
Test	2162,21	2162,32	97	3	0	0
3.	$\gamma_{var1} = 10$, $\beta_{var1} = 10$, $\beta_{var2} = 1$, $\beta_{age} = 1$ und $\gamma_{age} = 1$					
Train	2158,07	2158,08	100	0	0	0
Test	2162,55	2162,41	94	6	0	0
4.	$\gamma_{var1} = 10$, $\beta_{var1} = 1$, $\beta_{var2} = 1$, $\beta_{age} = 1$ und $\gamma_{age} = 1$					
Train	2155,77	2155,77	16	0	0	84
Test	2160,32	2160,29	16	0	0	84

Tabelle 19: Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable var1 über die 100 Seeds; Parameter γ_{var1} steuert die Korrelation während β_{var1} die Variable Importance bezüglich var1 reguliert

Die Variable var1 stammt aus einer Normalverteilung, die anderen Variablen dagegen nicht (siehe (20)). Und das ist der Grund, warum es keine erhebliche Veränderung gab. Um dies veranschaulichen zu können, wurden age, invests und var2 jeweils ebenso aus einer Normalverteilung gezogen, anstatt wie in (20) definiert. Die Ergebnisse in Tabelle 20 untermauern die Aussage, und deren RMSEs können den entsprechenden in den Tabellen 16, 17 und 18 gegenüber gestellt werden:

additive perturb.	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1. age	$\gamma_{age} = 1$, $\beta_{age} = 20$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2165,19	2165,22	2	0	0	98
Test	2169,18	2169,5	2	0	28	70
2. age	$\gamma_{age} = 10$, $\beta_{age} = 20$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2169,53	2169,52	4	0	0	96
Test	2174,36	2173,86	3	1	29	67
3. var2	$\gamma_{age} = 1$, $\beta_{age} = 1$, $\gamma_{var1} = 0$, $\beta_{var1} = 1$ und $\beta_{var2} = 10$					
Train	2158,97	2158,95	70	0	0	30
Test	2163,12	2163,23	64	6	5	25
4. invests	$\gamma_{age} = 10$, $\beta_{age} = 1$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2179,73	2179,87	100	0	0	0
Test	2184,03	2184,26	99	1	0	0

Tabelle 20: Variable Importance vs. Korrelation vs. RMSE hinsichtlich additiver perturbierter Variablen age, invests und var2 über die 100 Seeds; hier sind sie normalverteilt

Die letztgenannte Konfiguration wurde auch mit der Methode *correlated2* Noise Addition untersucht, deren Ergebnisse in Tabelle 21 erfasst sind. Es lässt sich feststellen, dass die ermittelten Werte sehr ähnlich zu den additiven in der vorherigen Tabelle 20 sind.

correlated2 perturb.	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1. age	$\gamma_{age} = 10$, $\beta_{age} = 20$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2169,53	2169,52	4	0	0	96
Test	2174,36	2173,92	2	2	28	68
2. var2	$\gamma_{age} = 1$, $\beta_{age} = 1$, $\gamma_{var1} = 0$, $\beta_{var1} = 1$ und $\beta_{var2} = 10$					
Train	2158,97	2158,95	70	0	0	30
Test	2163,12	2163,17	65	5	5	25

Tabelle 21: Variable Importance vs. Korrelation vs. RMSE hinsichtlich *correlated2* perturbierter Variable age bzw. var2 über die 100 Seeds; hier sind age und var2 normalverteilt

Der Vergleich dazu, wie die Werte aussehen, wenn die Variable nicht normalverteilt

ist, sondern wie in (20) beschrieben, ist in der folgenden Tabelle 22 dargestellt. Diese können auch mit den entsprechenden additiven Werten in Tabelle 18 verglichen werden.

correlated2 perturb. age	Mittelwert orig. RMSE	Mittelwert perturb. RMSE	varImp mit small RMSE	varImp mit large RMSE	non-varImp mit large RMSE	non-varImp mit small RMSE
1.	$\gamma_{age} = 10$, $\beta_{age} = 20$, $\gamma_{var1} = 1$, $\beta_{var1} = 1$ und $\beta_{var2} = 1$					
Train	2163,73	2176,74	0	100	0	0
Test	2168,41	2178,91	20	80	0	0

Tabelle 22: Variable Importance vs. Korrelation vs. RMSE hinsichtlich *correlated2* perturbierter Variable age über die 100 Seeds; hier stammt age nicht aus einer Normalverteilung, sondern aus einer Poisson Verteilung

Schließlich kann festgestellt werden: Ist die zu perturbierende Variable nicht normalverteilt, sind die Schlussforgerungen 1 bis 4 valide für beliebige Datensätze. Kommt sie jedoch aus einer Normalverteilung, sind keine erhebliche Steigerungen des RMSEs weder bei additiver noch bei *correlated2* Noise Addition zu erwarten. Anhand dieser Simulation konnten somit auch die festgestellten Punkte bei den im Fall betrachteten Ergebnissen in 1 belegt werden.

Zweite Simulation

Für die zweite Simulation wurden die für die erste Simulation generierten fünf verschiedenen Variablen (Y - Zielvariable - und vier Kovariablen) in der selben Art und Weise erstellt, auch aus den selben Verteilungen stammend mit jeweils 10.000 Beobachtungen. Die eben zu variierenden Parameter (wichtig für die erste Untersuchung) wurden hier alle auf 1 gesetzt und nicht mehr variiert. Viel mehr von Bedeutung ist nun die Untersuchung, ob sich der RMSE der Neuronalen Netzen über die simulierten Datensätze und Perturbation verändert oder nicht. Der Unterschied zur ersten Simulation ist zum einen, dass der Neuronale-Netze-Algorithums anstatt GBM verwendet wurde, zum anderen nur 10 verschiedenen Seeds (082022-082031) gesetzt wurden anstelle der vorherigen 100 und zum letzten nur Variable var2 perturbiert wurde. Bei all den 10 Seeds war der RMSE der perturbierten Daten exakt gleich der originalen, was somit Punkt 3 bestätigt.

6.1.3 Ergebnisse der Variable Importance

Hinsichtlich der Variable Importance mit (teilweise gleichzeitiger) und ohne Perturbation bei Random Forest und Gradient Boosting Machine bei den EUSILC Daten, sind folgende sechs Animationen erstellt worden.

Technischer Hinweis: Wenn diese PDF-Datei mit dem Adobe Programm geöffnet

wird, laufen die folgende und die weiteren Animationen automatisch über die verschiedenen verwendeten Anonymisierungsverfahren hinweg.

Bei perturbierten Daten mit Microaggregation (mit den diversen aggregierten *k*s) scheint es sowohl bei Gradient Boosting Machine als auch bei Random Forest keinen großen Unterschied hinsichtlich der Variable Importance zu geben (siehe 17 und 20). Anders ist es allerdings bei sowohl additiver (siehe 18 und 21) als auch *correlated2* Noise Addition (siehe 19 und 22). Da verlieren die perturbierten Variablen sehr schnell an Bedeutung. Am deutlichsten ist es beim additiven Fall.

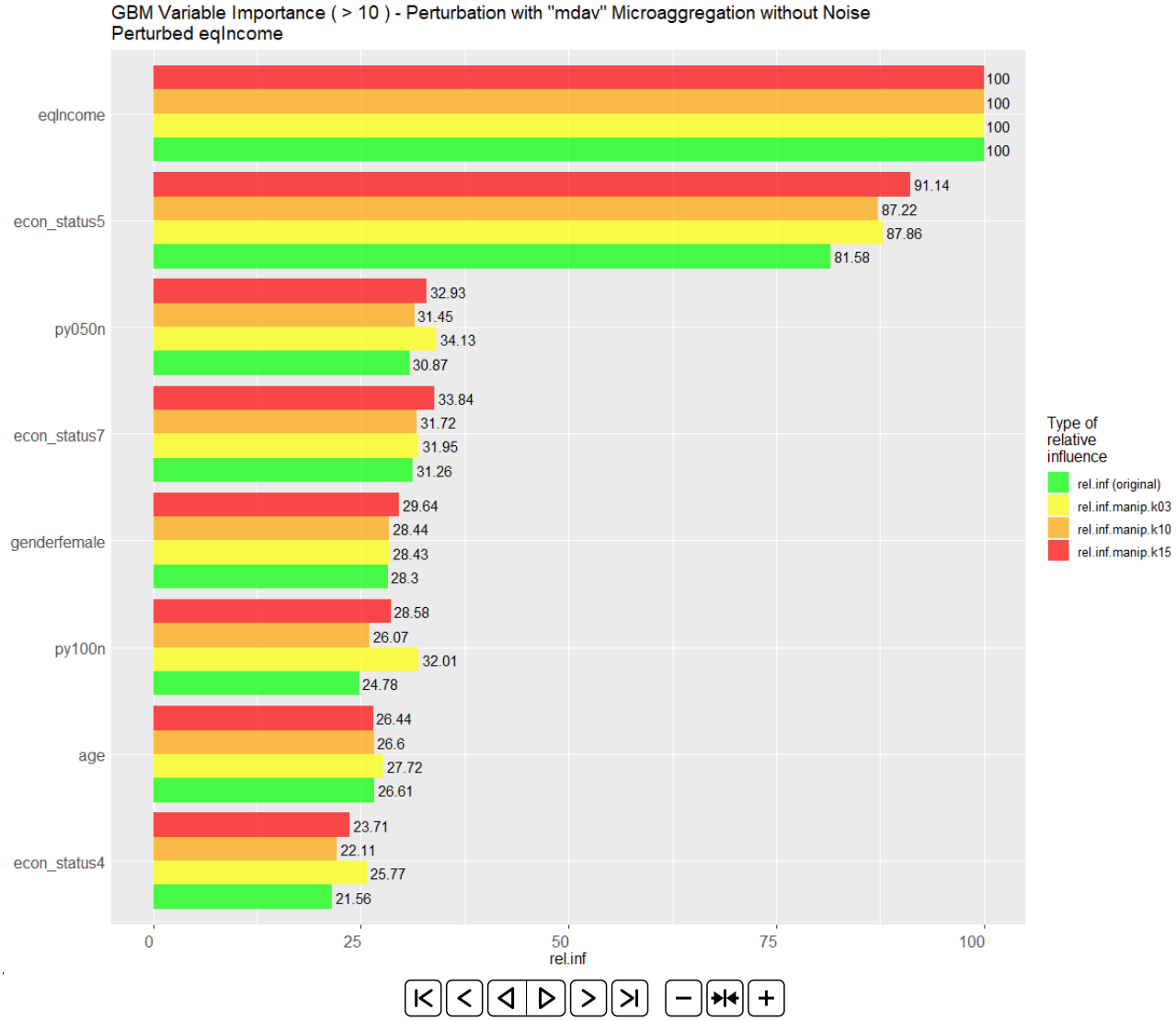


Abbildung 17: Animierter Vergleich der Variable Importance (> 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei MDAV Microaggregation (ohne Noise)

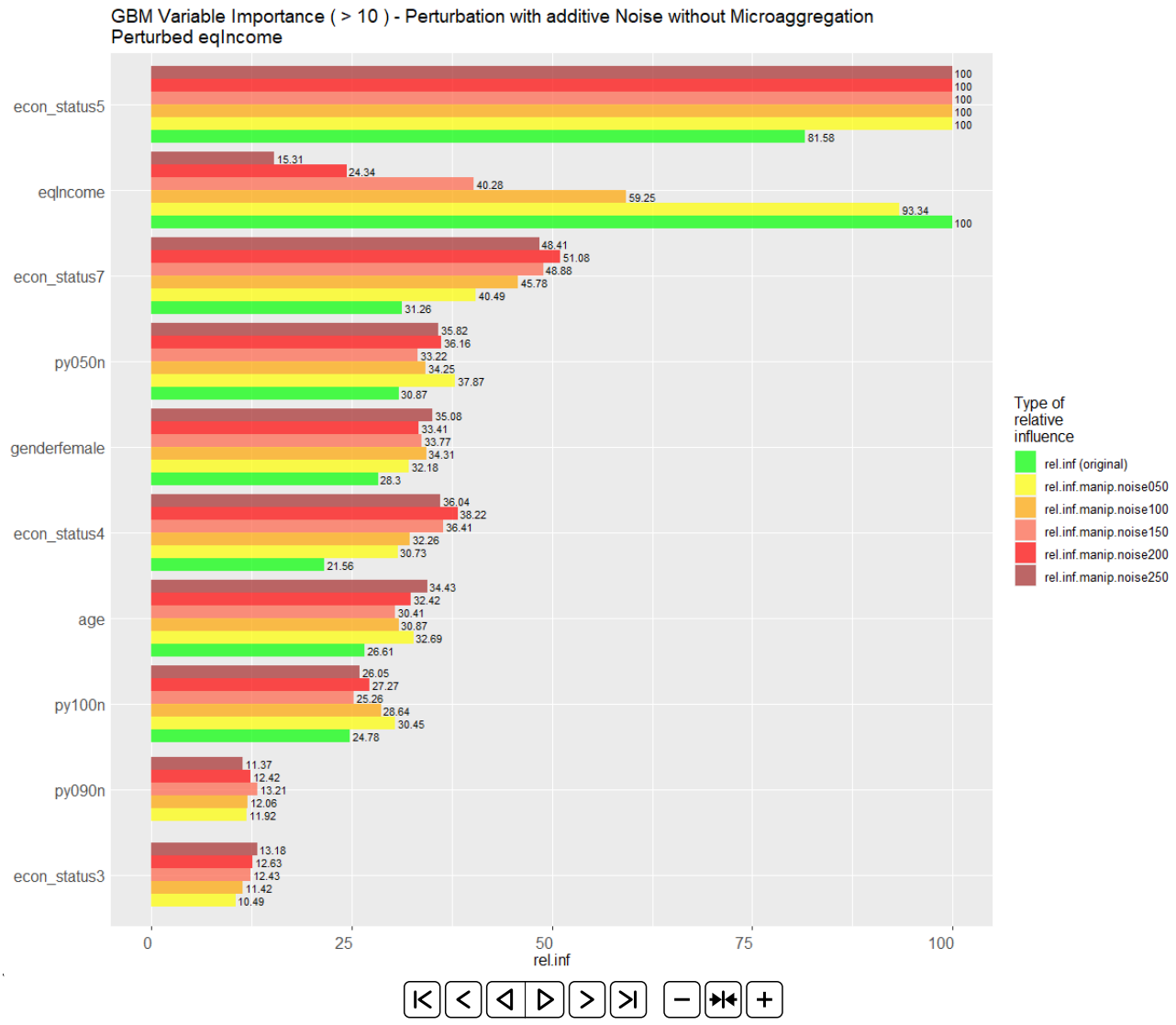


Abbildung 18: Animierter Vergleich der Variable Importance ($>$ 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *additive* Noise (ohne Microaggregation)

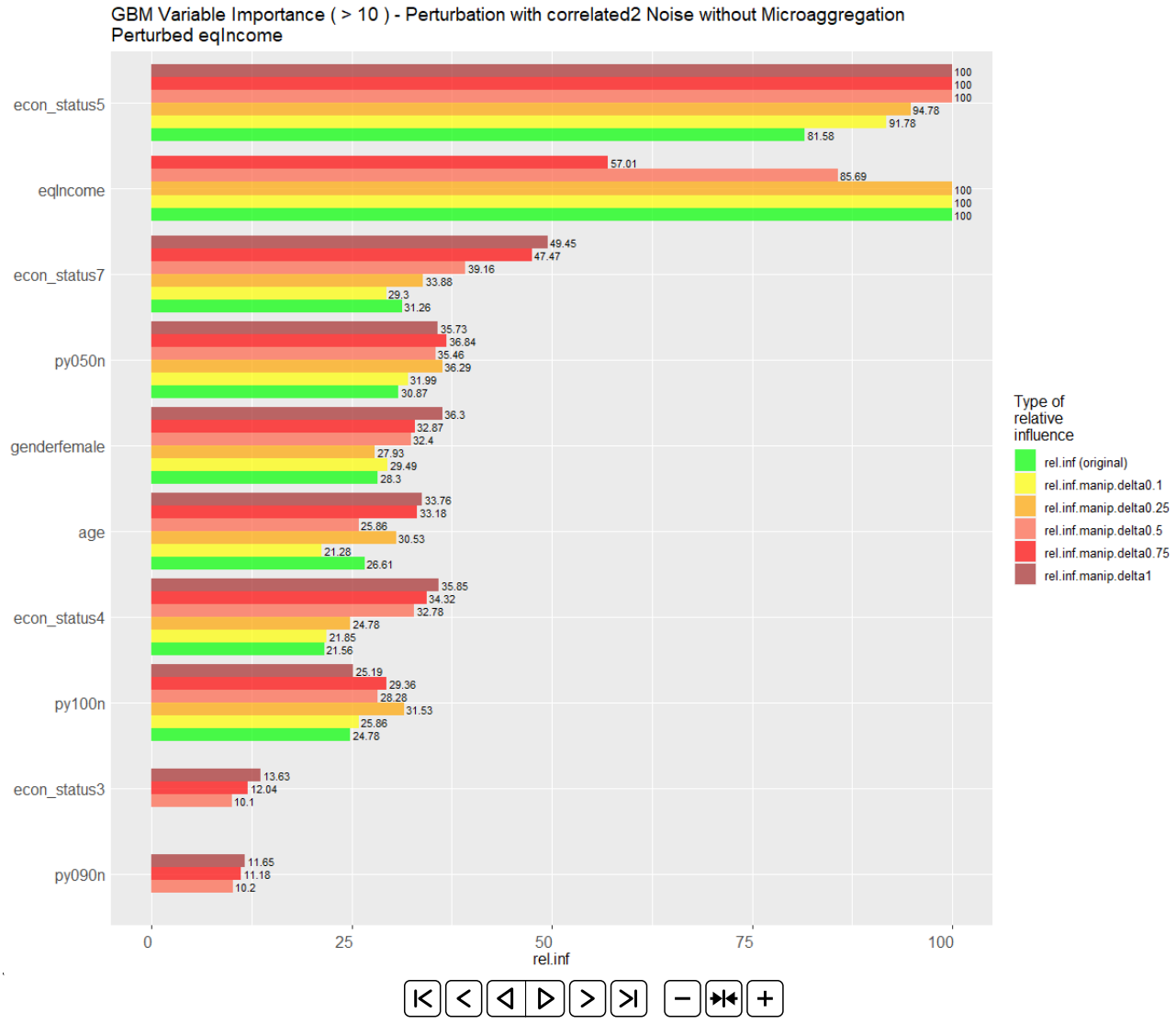


Abbildung 19: Animierter Vergleich der Variable Importance (> 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *correlated2* Noise (ohne Microaggregation)

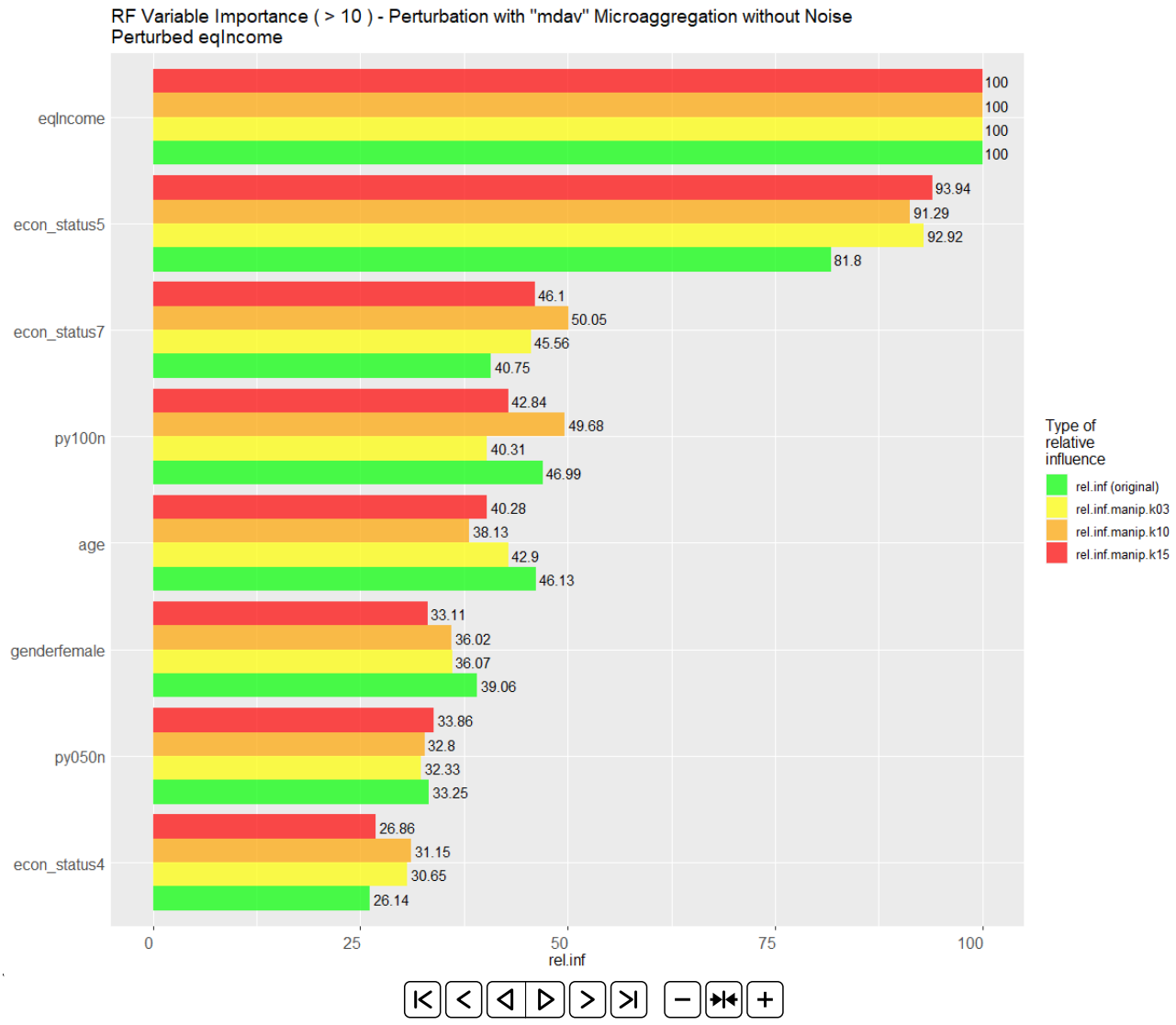


Abbildung 20: Animierter Vergleich der Variable Importance (> 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei MDAV Microaggregation (ohne Noise)

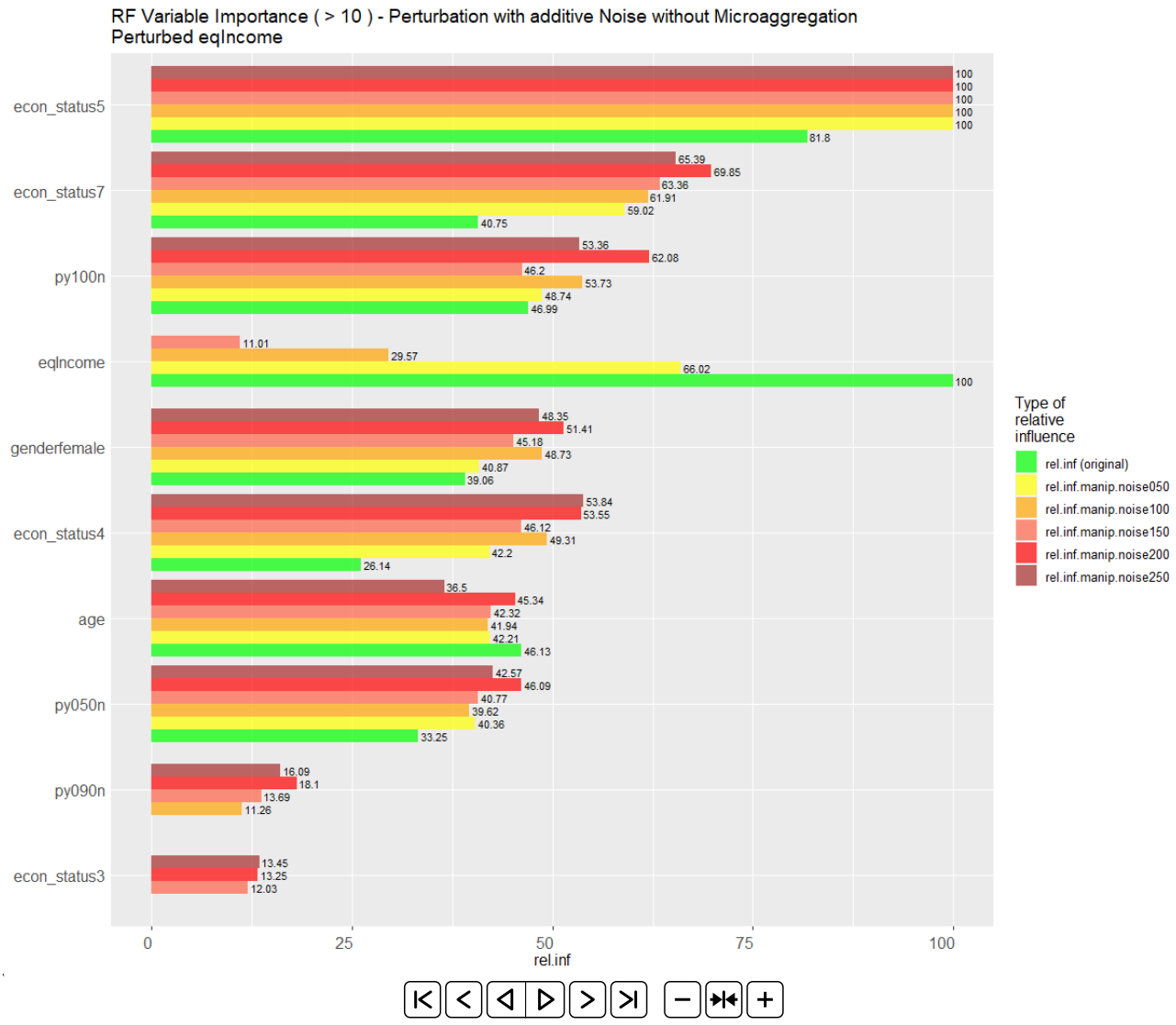


Abbildung 21: Animierter Vergleich der Variable Importance ($>$ 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *additive* Noise (ohne Microaggregation)

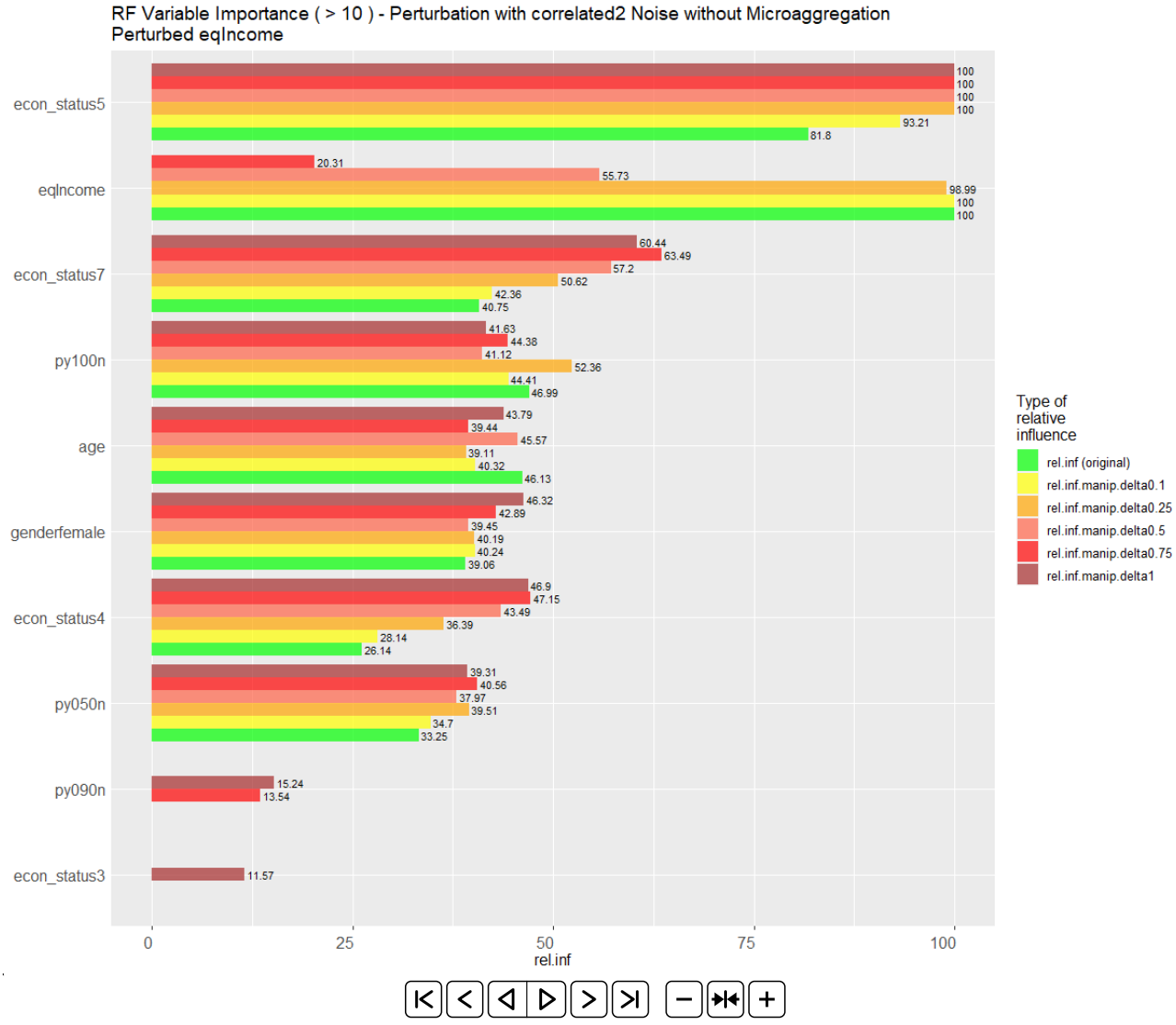


Abbildung 22: Animierter Vergleich der Variable Importance (> 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *correlated2* Noise (ohne Microaggregation)

6.1.4 Ergebnisse Offenlegungsrisiko vs. Informationsverlust

Eine weitere Simulation wurde aufgebaut und durchgeführt, um das Offenlegungsrisiko dem Informationsverlust gegenüber zu stellen. Die ursprüngliche Idee war zunächst zu überprüfen, ob sich das, was Benschop and Welch (2019) in 3.2.1 hinsichtlich Microaggregation behauptet haben, hier auch widerspiegelt. Anschließend wurde die Simulation mit zwei weiteren Anonymisierungstools erweitert.

Hier wurde die Simulation 500 Mal wiederholt, mit 500 verschiedenen Seeds (082022-082521), wobei bei jedem Seed das Offenlegungsrisiko (in % und gemäß 2.6 mittels Intervalloffenlegung) sowie der Informationsverlust (mittels IL1s als Metrik definiert in 2.7)

und von beiden zum Schluss der Mittelwert berechnet wurden. Des Weiteren wurden drei Anonymisierungsverfahren verwendet: 1) Microaggregation mit Aggregationslevel 3 und Methode MDAV; 2) *additive* Noise mit Menge an Noise = 250; 3) *correlated2* Noise mit delta = 0,5. Ferner wurden die selben Variablen mit den gleichen Verteilungen wie in (20) definiert. Zudem sind noch zwei zusätzliche kategoriale Variablen aus dem EUSILC Datensatz hinzugekommen, nämlich db040 (= fed_state) und rb090 (= gender), welche für die Bestimmung der Key Variablen im SDC-Objekt in R dienen sollten. Die perturbierten Variablen waren nämlich var1, var2, invests und age. Die festzulegenden Parameter, wie in (20) etabliert, sind bei einigen konstant geblieben (d.h. $\beta_{var1} = 1$, $\beta_{age} = 1$ und $\beta_{var2} = 1$, welche in den vorherigen Simulationen die Variable Importance gesteuert hatten), während sie bei den restlichen (γ_{var1} - steuert Korrelation zwischen var1 und var2 - und γ_{age} - steuert Korrelation zwischen invests und age) variiert wurden. Sind Letztgenannte gleich 1, sind die Variablen kaum korreliert. Wenn sie dagegen gleich 10 sind, weisen die Variablen eine starke Korrelation auf.

Zunächst sind alle vier Variablen gleichzeitig perturbiert worden, deren Resultate in folgender Tabelle 23 veranschaulicht sind. Daraus lässt sich ablesen, dass je stärker die Korrelation zwischen den Variablen bei der Microaggregation, desto kleiner ist der Informationsverlust, was Benschop and Welch (2019) in 3.2.1 behauptet haben. Dies trifft dennoch nur bei Microaggregation zu; bei *additive* sowie *correlated2* Noise Addition bleibt das Verhältnis nahezu unverändert.

perturbierte Variablen: invests, age, var1 und var2								
	$\gamma_{var1} = 1$, $\gamma_{age} = 1$		$\gamma_{var1} = 1$, $\gamma_{age} = 10$		$\gamma_{var1} = 10$, $\gamma_{age} = 1$		$\gamma_{var1} = 10$, $\gamma_{age} = 10$	
	DR (%)	IL	DR (%)	IL	DR (%)	IL	DR (%)	IL
Microag.	6,74	2.107,52	7,77	2.068,15	6,82	2.020,41	7,81	1.979,91
AddNoise	0,0003	56.410,41	0,0003	56.410,41	0,0003	56.410,41	0,0003	56.410,41
Corr2Noise	0,0032	11.676,58	0,0033	11.678,68	0,0033	11.676,15	0,0034	11.678,25

Tabelle 23: Disclosure Risk (DR) vs. Informationsverlust (IL) bezüglich drei Anonymisierungsverfahren auf den Variablen mit unterschiedlichen Korrelationsstrukturen; simultane Perturbation aller vier Variablen

Anschließend erfolgte eine simultane gruppenweise Perturbation mit Rücksicht auf die Korrelation zwischen den Variablen, indem zunächst nur die Gruppe invests und age simultan kontaminiert wurde, während die anderen Variablen unperturbiert geblieben sind. Dann wurde der umgekehrte Fall durchgespielt (d.h. nur var1 und var2 wurden perturbiert, die anderen nicht). Tabelle 24 stellt die Ergebnisse dazu dar, die denen in 23 gegenübergestellt werden können. Die andere Behauptung von Benschop and Welch (2019) in 3.2.1 lässt sich dadurch ebenso belegen: Die simultane Perturbation ausschließlich innerhalb korrelierter Gruppen bzw. Variablen anstelle von allen verringert den Informa-

tionsverlust, sogar bei den drei angewendeten Anonymisierungsverfahren. Dazu ist aber ebenfalls zu beachten, dass bei der Microaggregation das Offenlegungsrisiko (DR) auch extrem gesteigert ist, sodass es mit diesem hohen Risiko wahrscheinlich nicht praktikabel ist, die Daten zu veröffentlichen.

	perturbierte Variablen: invests, age		perturbierte Variablen: var1, var2	
	$\gamma_{var1} = 1$, $\gamma_{age} = 10$		$\gamma_{var1} = 10$, $\gamma_{age} = 1$	
	DR (%)	IL	DR (%)	IL
Microag.	96,58	72,49	96,44	131,24
AddNoise	0,1850	28.202,1	0,1850	28.202,1
Corr2Noise	0,5947	5.838,94	0,5898	5.839,47

Tabelle 24: Disclosure Risk (DR) vs. Informationsverlust (IL) bezüglich drei Anonymisierungsverfahren auf den Variablen mit unterschiedlichen Korrelationsstrukturen; gruppenweise Perturbation innerhalb der korrelierten Gruppe

6.1.5 (β)-Korrekturansatz bei metrischer X^* mittels SIMEX Methode

Dieses Unterkapitel beschäftigt sich mit einem Korrekturansatz hinsichtlich β bei metrischer, mit additivem Fehler perturbierter Kovariable X^* in einem linearen Regressionsmodell. Carroll et al. (2006, S. 1) zufolge haben Messfehler in den Kovariablen drei Effekte ("Whammy Effects"): 1) Sie verursachen Bias bei der Schätzung der Parameter in statistischen Modellen; 2) Sie führen zu einem (schwerwiegenden) Verlust an Power bei der Detektion der Beziehungen zwischen den Variablen; 3) Sie verdecken die Merkmale der Daten, was die grafische Modellanalyse beeinträchtigt. Diese Effekte werden in einem aus Carroll et al. (2006, S. 2) entnommenen Beispiel illustriert:

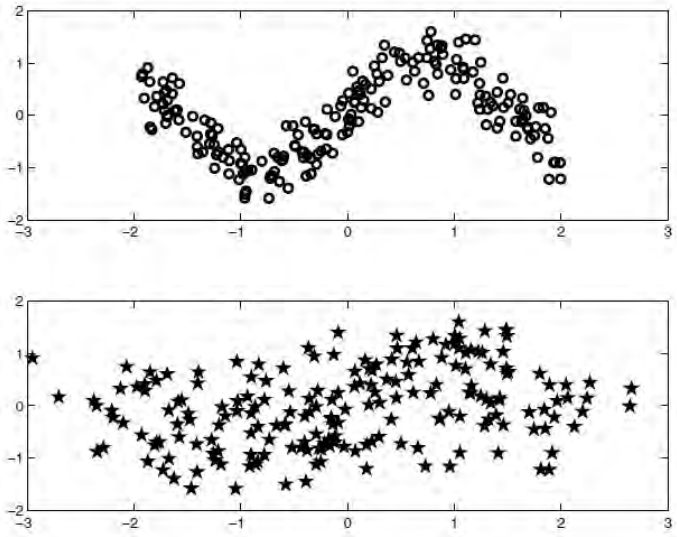


Abbildung 23: Whammy-Effekte der Messfehler in den Kovariablen (aus Carroll et al. (2006, S. 2) entnommen); der oberste Plot stellt die wahre Variable dar, der unterste dagegen die perturbierte

Um das Ganze zu formalisieren, wird eine einfache Lineare Regression für jede Beobachtung i mit wahrer Variable X

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

angenommen (ähnlich zu 4.2.1), wobei β_1 folgendermaßen geschätzt werden kann (siehe z.B. Yi (2017, S. 45)):

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

wobei $\bar{X}$ und $\bar{Y}$ die Mittelwerte von jeweils X und Y sind.

Wenn die Kovariable X jedoch perturbiert ist, wird sie durch X^* in der obigen Formel ersetzt:

$$\hat{\beta}_1^* = \frac{\sum_{i=1}^{n}(X_i^* - \bar{X}^*)(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i^* - \bar{X}^*)^2}$$

wobei hier ein klassischer additiver Fehler für $X_i^* = X_i + u_i$ (mit $u \sim N(0, \sigma_u^2)$, $X \perp u$ und $u \perp \varepsilon$) angenommen wird. Das führt zu einem Attenuation Effekt (Abschwächungseffekt) bei der Schätzung des β_1 Koeffizienten, da $\hat{\beta}_1^*$ in Wahrscheinlichkeit zu $\hat{\beta}_1^*$ (anstatt zu $\hat{\beta}_1$) konvergiert, wenn $n \to \infty$, was $\hat{\beta}_1^*$ somit zu keinem konsistenten Schätzer für β_1 macht. Dieser Attenuation Effekt ist wie folgt definiert (siehe z.B. Yi (2017, S. 46)):

$$\hat{\beta}_1^* = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_u^2}\beta_1 = \frac{var(X_i)}{var(X_i^*)}\beta_1$$

und in Abbildung 24 dargestellt:

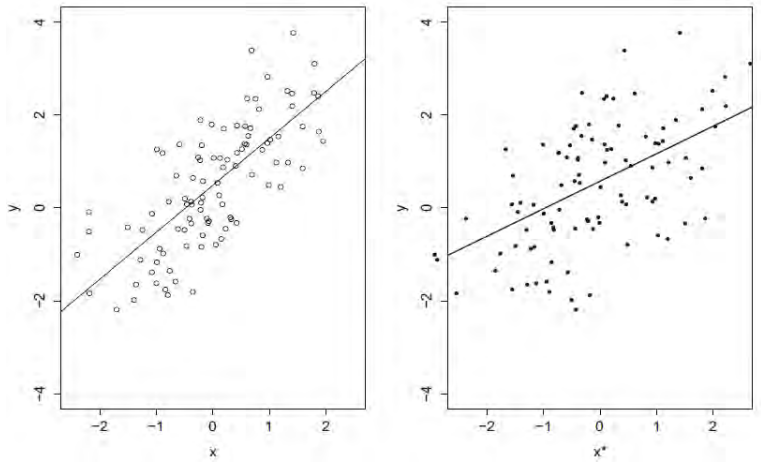


Abbildung 24: Darstellung des Attenuation Effekts bei einer perturbierten Kovariablen X^* (aus Yi (2017, S. 47) entnommen)

An dieser Stelle mag die Frage auftreten: Wie können die $\hat{\beta}^*$ Koeffizienten dennoch korrigiert werden, um die wahre Beziehung zwischen X und Y zu erhalten?

Eine vorgeschlagene Korrektur in dieser Hinsicht ist u.a. die von Cook and Stefanski (1994) entwickelte SIMEX Methode, deren Prozedur in zwei Schritte aufgeteilt ist: der Simulation- und der Extrapolation-Step. Diese Methode kann bei einem Regressionsmodell immer angewendet werden, unabhängig davon, ob ein Lineares oder allgemeines nicht Lineares Modell zugrunde liegt. Im Folgenden wird sie beschrieben und an einem Linearen Modell illustriert bzw. angewendet.

Die Idee dabei ist, künstliche Fehler auf die Daten in zunehmender Höhe hinzuzufügen, dann eine Beziehung zwischen der Höhe des Fehlers und den naiven Koeffizienten (d.h. mit Ignorierung des Messfehlers) zu schätzen und abschließend die wahren Koeffizienten (Messfehler Null) zu extrapolieren (vgl. Lederer and Küchenhoff, 2006, S. 26). Aus diesem Anlass wird folgende Funktion definiert (Lederer and Küchenhoff, 2006, S. 27):

$$\sigma_u^2 \to \beta^*(\sigma_u^2) := \mathcal{G}(\sigma_u^2)$$

wobei β^* das Limit ist, zu dem der naive Schätzer konvergiert, wenn $n \to \infty$, und $\mathcal{G}(\sigma_u^2) = 0$ die Konsistenz des Schätzers bezeichnet. Die SIMEX Methode versucht die obige Funktion $\mathcal{G}(\sigma_u^2)$ mit einer parametrischen Funktion $\mathcal{G}(\sigma^2, \Gamma)$, wie z.B. einer quadratischen Funktion

$$\mathcal{G}_{quad}(\sigma_u^2, \Gamma) = \gamma_0 + \gamma_1 \sigma_u^2 + \gamma_2 (\sigma_u^2)^2 \tag{21}$$

zu approximieren (idem).

Die folgenden Erläuterungen zum Simulation- bzw. Extrapolation-Step basieren auf Lederer and Küchenhoff (2006, S. 27), auf Shaw (2017, S. 58-62) sowie auf Cook and Stefanski (1994).

Beim Simulation-Step wird Γ geschätzt, was in drei Teile aufgesplittet wird. Für $b = 1, ..., B$ Bootstrap Iterationen, mit bekanntem σ_u^2, festzulegendem $\lambda > 0$ (Menge des zu addierenden Messfehlers) und $W_{b,i} \sim N(0, 1)$:

1. Generiere $X_{b,i} = X_i^* + \sqrt{\lambda}\sigma_u W_{b,i}$. Somit wird die gesamte Varianz von $X_{b,i} = (1+\lambda)\sigma_u^2$.
2. Fitte ein Lineares Regressionsmodell mit X^* ersetzt durch die gerade generierte $X_{b,i}$ und schätze $\beta_{b,\lambda}$.
3. Berechne $\hat{\beta}_\lambda$ als den Durchschnitt über alle B Schätzungen von $\beta_{b,\lambda}$, was der Schätzung von $\mathcal{G}((1 + \lambda)\sigma_u^2)$ entspricht.

Lederer and Küchenhoff (2006, S. 27) zufolge führt die Durchführung dieser Iterationen für ein festes Grid von λs zu einem Schätzer für $\hat{\Gamma}$ in $\mathcal{G}(\sigma_u^2, \Gamma)$. Eine gute Wahl für dieses Grid ist $\lambda \in (0.5, 1, 1.5, 2)$ (idem).

Anschließend erfolgt der Extrapolation-Step, bei dem $\mathcal{G}(\sigma_u^2, \Gamma)$ zum Fall Messfehler Null (d.h. $\sigma_u^2 = 0$) wie folgt zurück extrapoliert wird:

1. Fitte eine Kurve für die erhaltenen bzw. geschätzten Paare $(\lambda, \hat{\beta}_\lambda)$.

2. Extrapoliere den Prozess zurück, indem $\lambda = -1$ gesetzt wird, was somit den SIMEX Schätzer (d.h. $\hat{\beta}_{SIMEX}$) letztendlich ausmacht, denn $\hat{\beta}_{SIMEX} := \mathcal{G}(0, \Gamma)$.

Um die Effektivität der SIMEX Methode im Rahmen einer Korrektur von β bei perturbierter metrischer Kovariable X^* zu untersuchen bzw. auszunutzen, wurde eine Simulation in `R` mit folgendem Linearen Regressionsmodell konstruiert:

$$Y = \beta_0 + 1.3X + 3.2Z + \varepsilon \tag{22}$$

wobei $X \sim N(\mu = 10, \sigma^2 = 3^2)$ bzw. $X \sim Pois(\lambda = 7)$, $Z \sim N(\mu = 6, \sigma^2 = 2.5^2)$ und $\varepsilon \sim N(\mu = 0, \sigma^2 = 1)$. Ferner wurde eine perturbierte Version aus (22) generiert, indem X durch X^* ersetzt wurde, mit $X^* = X + u$, $u \sim N(\mu = 0, \sigma^2 = \sigma_u^2)$, $X \perp u$ und $u \perp \varepsilon$. Die Simulation wurde 500 Mal wiederholt, mit 500 unterschiedlichen Seeds (102022-102521) und aus den verschiedenen Parametern, die durch das Modell geschätzt werden, wurde der Mittelwert berechnet. Als eine Approximation für $\mathcal{G}(\sigma_u^2, \Gamma)$ wurde die in (21) dargestellte quadratische Funktion herangezogen. Zudem wurden die Hyperparameter n und σ_u variiert, d.h. $n \in \{100, 1000\}$ und $\sigma_u \in \{0.5, 2\}$. Bei X wurde auch untersucht, ob es einen Unterschied ausmacht, wenn die Variable normalverteilt ist oder nicht. Dafür wurde die Option "normal = TRUE" eingebaut. Falls es nicht gewünscht ist, stammt X aus der obigen Poisson Verteilung. Die Anzahl an B Bootstrap Iterationen wurde auf 500 sowie das Grid für $\lambda \in \{0.5, 1, 1.5, 2\}$ festgelegt.

Mit all diesen Konfigurationen wurden drei Modelle gefittet: 1) das wahre Modell (stammend aus (22)); 2) das naive Modell (bei dem die Messfehler der Variable X^* ignoriert werden); 3) das SIMEX Modell. Die Ergebnisse der Simulation sind in der Tabelle 25 dargestellt.

Aus dieser Tabelle lässt sich erkennen, dass das naive Lineare Regressionsmodell ungeeignet bzw. verzerrt für die Schätzung des Koeffizienten für Variable X bei all den getesteten Konfigurationen ist. Besonders verzerrt ist es im Falle einer hohen Menge an Messfehlern (wie z.B. $\sigma_u = 2$), bei dem der naive Schätzer für X einen Koeffizienten von circa 0.90 (unter Normalverteilung) bzw. 0.82 (unter Poisson Verteilung) schätzt, dessen Wert weit entfernt vom wahren 1.30 ist. Auch das SIMEX Modell ist von diesem Manko ein wenig betroffen, allerdings nicht so stark, nämlich circa 1.175 (unter Normalverteilung) bzw. 1.115 (unter Poisson Verteilung). Wenn der addierte Messfehler dennoch klein ist (wie z.B. $\sigma_u = 0.5$), erreicht die SIMEX Methode eine konsistente Schätzung für β_{X^*}. Ferner wurde für die nicht kontaminierte Variable Z bei allen Varianten ebenso eine konsistente Schätzung für β_Z erhalten.

(Hyper)Parameter			X normalverteilt			X Poisson verteilt		
n	σ_u	β	True	Naiv	SIMEX	True	Naiv	SIMEX
100	0.5	β_0	-0.01858	0.33638	-0.01782	-0.00282	0.31593	0.00347
100	0.5	X(*)	1.30194	1.26711	1.30242	1.30128	1.25634	1.30101
100	0.5	Z	3.19969	3.19841	3.19851	3.19916	3.19873	3.19858
100	2	β_0	-0.01858	4.03371	1.27699	-0.00282	3.32523	1.30182
100	2	X(*)	1.30194	0.90005	1.17514	1.30128	0.82720	1.11714
100	2	Z	3.19969	3.19434	3.19473	3.19916	3.19975	3.19793
1000	0.5	β_0	-0.00134	0.34349	-0.00713	0.00384	0.32270	0.01157
1000	0.5	X(*)	1.29991	1.26571	1.30077	1.29915	1.25401	1.29845
1000	0.5	Z	3.20022	3.20009	3.20012	3.19980	3.19959	3.19961
1000	2	β_0	-0.00134	3.99335	1.24290	0.00384	3.32765	1.30673
1000	2	X(*)	1.29991	0.90134	1.17651	1.29915	0.82566	1.11445
1000	2	Z	3.20022	3.19957	3.19982	3.19980	3.19892	3.19907

Tabelle 25: Vergleich der Ergebnisse der Koeffizienten mittels True-, Naiv- und SIMEX Modell mit X normalverteilt vs. nicht

Als Illustration für den Extrapolation-Step mit der oben definierten quadratischen Approximation beim SIMEX Modell (mit $n = 1000, \sigma_u = 2,$, normal = FALSE und Seed = 102023) wurde folgende Abbildung 25 erstellt:

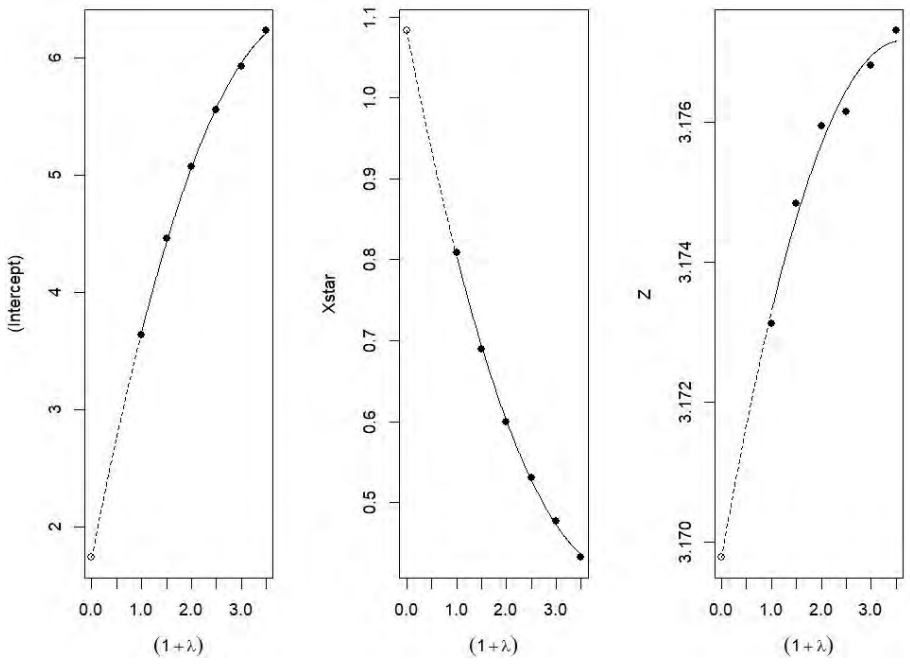


Abbildung 25: Darstellung des Extrapolation-Steps für das SIMEX Modell mit einer quadratischen Approximation

6.1.6 Korrekturansatz für individuelle Prädiktionen y_i bei metrischer X^*

In diesem Abschnitt wird ein Korrekturansatz für die individuellen Prädiktionen y_i bei metrischer, mit additivem Fehler perturbierter Kovariable X^* in einem quadratischen Regressionsmodell diskutiert. Eine dazu angemessene Korrektur ist die Regressionskalibrierung, welche für alle Arten von Regressionsmodellen angewendet werden kann. Die Idee dahinter ist, anstelle der Aufnahme der perturbierten Kovariable X^* den bedingten Erwartungswert der wahren bedingt auf die perturbierte Kovariable (d.h. $\mathbb{E}(X|X^*)$) in das Regressionsmodell aufzunehmen (siehe z.B. Yi (2017, S. 60)). Was bringt dieses Vertauschen der Daten mit sich? Die perturbierte Kovariable X^* trägt zu einer Verbesserung der Prognose der wahren Kovariable X bei.

Um dieses Korrekturverfahren auf Beobachtungsebene darzulegen, wird ein additiver, von X unabhängiger Messfehler u_i auf eine wahre Kovariable X_i für Beobachtung $i = 1, ..., n$ addiert:

$$X_i^* = X_i + u_i \tag{23}$$

Zunächst wird angenommen, die Kovariable X in 23 sowie der Messfehler u seien normalverteilt, mit $X \sim N(\mu_X, \Sigma_X)$ und $u \sim N(0, \Sigma_u)$. Daraus folgt:

$$A \begin{pmatrix} X_i \\ u_i \end{pmatrix} = \begin{pmatrix} X_i \\ X_i^* \end{pmatrix} \tag{24}$$

wobei $A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ ist. Ferner ist dieser Vektor $(X_i, X_i^*)^t$ bivariat normalverteilt, mit $\mu = A(\mu_{X_i}, \mu_{u_i})^t$ und $\Sigma = A(\Sigma_{X_i}, \Sigma_{u_i})^t A^t$ (siehe z.B. Fahrmeir et al., 2015, S. 26). Des Weiteren wird dessen bedingter Erwartungswert benötigt (siehe z.B. Fahrmeir et al., 2015, S. 27):

$$\mathbb{E}(X_i|X_i^*) = \mu_{X_i|X_i^*} = \mu_{X_i} + \Sigma_{(XX^*)_i}\Sigma_{X_i^*}^{-1}(x_i^* - \mu_{X_i^*}) \tag{25}$$

Ist $\mathbb{E}(X|X^*)$ berechnet worden, ersetzt er die Kovariable X^* bei dem herangezogenen Regressionsmodell, um Y zu prognostizieren.

Dieses Regressionskalibrierung Verfahren wurde in `R` mittels einer Simulation untersucht. Dabei wurden fünf verschiedene Variablen (Y und vier Kovariablen) aus verschiedenen Verteilungen stammend mit jeweils 10.000 Beobachtungen generiert (siehe 26).

$$\begin{cases} var1 \sim N(\mu = 88,\ \sigma^2 = 5^2), \\ var2 \sim N(\mu = 40,\ \sigma^2 = 14^2), \\ var3 \sim U(min = 10, max = 200), \\ var4 \sim \big(Par(n = 7.500, x = 1.500, \alpha = 4) + (n = 2.500, x = 0)\big), \\ Y = var1 + (var2)^2 + 2 \cdot var3 + 0,4 \cdot var4 + \varepsilon, \end{cases} \tag{26}$$

wobei $\varepsilon \sim N(0, 10)$ ist. Die Formula für das Training und Testen des Algorithmus ist wie folgt definiert:

$$Y \sim var1 + var2 + I(var2)^2 + var3 + var4 \tag{27}$$

Anschließend erfolgte der (80:20) Split in Trainings- und Testdaten. Dazu wurde ein 100-facher GBM-Algorithmus auf die originalen Daten angewendet und daraus der Mittelwert und Median berechnet. Diese Prozedur wurde zudem 100 Mal wiederholt, mit 100 verschiedenen festgelegten Seeds (102022-102121), sodass 100 neue Datensätze erzeugt wurden.

Als Nächtes wurde eine perturbierte Version für $var2$ aus (26) generiert, indem Letztgenannte durch $var2^*$ ersetzt wurde, mit $var2^* = var2 + u$, $u \sim N(\mu = 0, \sigma^2 = 5^2)$, $var2 \perp u$ und $u \perp \varepsilon$. Danach wurden analog, neben dem wahren bereits trainierten Modell (stammend aus (26)), zwei weitere gefittet: Ein naives Modell (bei dem die Messfehler der Kovariable $var2^*$ ignoriert wurden) und ein Modell, in dem die Regressionskalibrierung mittels (24) und (25) erfolgte. Außerdem wurde untersucht, ob es einen Unterschied beim RMSE gibt, wenn die $var2$ normal- oder Poisson verteilt ist. Auch ein Lineares anstelle eines quadratischen Regressionsmodells wurde ausprobiert. Dabei wurde $Y = 0,3 \cdot var1 + 100 \cdot var2 + 5 \cdot var3 + 0,9 \cdot var4 + \varepsilon$ (statt wie in (26)) definiert, um zu gewährleisten, dass $var2$ wichtig wäre. Zudem ändert sich die Formula in (27) dahingehend, dass der quadratische Term $I(var2)^2$ wegfällt. Die Ergebnisse der Simulation sind in der Tabelle 26 dargestellt:

RMSE		$var2$ normalverteilt			$var2$ Poisson verteilt		
Regression	Daten	True	Naiv	RC	True	Naiv	RC
quadratisch	train	134.29	134.36	134.38	120.85	160.61	160.61
quadratisch	test	150.2	149.64	150.19	138.56	175.78	175.78
linear	train	144.59	144.65	144.65	122.36	155.66	155.66
linear	test	158.67	158.54	158.54	140.38	169.29	169.29

Tabelle 26: Vergleich der Ergebnisse des RMSEs mittels True-, Naiv- und RC (Regressionskalibrierung) Modell mit $var2$ normalverteilt vs. nicht und quadratische vs. lineare Regressionsart

Daraus lässt sich leider konstatieren, dass weder beim Linearen noch beim quadratischen Regressionsmodell die Regressionskalibrierung einen Mehrwert für die Verbesserung der Vorhersage im Vergleich zum naiven Modell brachte. Dass eine normalverteilte Kovariable im naiven Modell keinen erheblichen RMSE verursacht, wurde schon hier 6.1.2 festgestellt. Auf Grund dessen besteht hierzu ein Anlass für zukünftige Untersuchungen, welche nach möglichen Lösungen bzw. Alternativen in dieser Hinsicht suchen.

6.2 Binäre Zielvariable

In diesem Unterkapitel werden Korrekturansätze für die Parameter βs präsentiert (siehe 6.2.1 und 6.2.2). Darüber hinaus wurden zwei Korrekturmethoden entwickelt, welche die individuellen Prädiktionen y_i zu adjustieren versuchen, wenn zum einen eine binäre Kovariable X^* sowie zum anderen eine binäre Zielvariable Y^* perturbiert wurde, um eine höhere Accuracy zu erhalten (siehe 6.2.3 und 6.2.4).

6.2.1 (β)-Korrekturansatz für Y^* mittels Newton-Raphson-Algorithmus

Im Folgenden wird ein Korrekturansatz in einer Logistischen Regression deskribiert, in der die Zielvariable mittels PRAM kontaminiert wurde (siehe 5.2.2). Dieser Korrekturansatz, nämlich mittels Newton-Raphson-Algorithmus, stützt sich teilweise auf van den Hout (1999, S. 63). Nachfolgend (in 6.2.1) wird auch eine ausführliche Herleitung dessen Ergebnisse gezeigt. Ferner wird bei der Methode die Annahme getroffen, dass die Transitions- bzw. Markov Matrix mit den perturbierten Daten von den NSIs veröffentlicht wird. Trifft dies nicht zu, benötigt der Korrekturansatz eine zusätzliche Schätzung der $\boldsymbol{P}_Y$ Matrix bzw. muss er angepasst werden.

Dieser Newton-Raphson-Algorithmus benötigt die Score- und Fisher Information-Funktionen wie in 5.2.1 definiert, welche aus (16) abgeleitet werden können, um iterierend das optimalste β zu erreichen. Des Weiteren kann π wie folgt bestimmt werden (siehe (7)):

$$\pi = P(Y = 1|X) = \frac{\exp(\beta^t X)}{1 + \exp(\beta^t X)}$$

Die (perturbierte) Score Funktion lautet somit (siehe Rechenweg auf der übernächsten Seite 6.2.1):

$$
\begin{aligned}
S^*(\beta) = \frac{\delta}{\delta\beta_j} l^*(\beta) &= \sum_{i=1}^{n} \Bigg\{ y_i^* \left[\frac{\delta}{\delta\beta_j} \log\Bigg(p_{01} + (p_{11} - p_{01}) \Big(\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \Big) \Bigg) \right] + \\
&\quad (1 - y_i^*) \left[\frac{\delta}{\delta\beta_j} \log\Bigg(p_{00} + (p_{10} - p_{00}) \Big(\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \Big) \Bigg) \right] \Bigg\} \\
&= \sum_{i=1}^{n} \Bigg\{ y_i^* \left[\frac{x_{ij} \cdot (p_{11} - p_{01}) \cdot \exp(\beta^t x_i)}{\Big(\exp(\beta^t x_i) + 1 \Big) \Big(p_{11} \cdot \exp(\beta^t x_i) + p_{01} \Big)} \right] + \\
&\quad (1 - y_i^*) \left[\frac{x_{ij} \cdot (p_{10} - p_{00}) \cdot \exp(\beta^t x_i)}{\Big(\exp(\beta^t x_i) + 1 \Big) \Big(p_{10} \cdot \exp(\beta^t x_i) + p_{00} \Big)} \right] \Bigg\}
\end{aligned} \tag{28}
$$

und die (perturbierte) Fisher Information Funktion lautet:

$$
\begin{aligned}
I^*(\beta) &= -\frac{\delta^2}{\delta\beta_j \delta\beta_s} l^*(\beta) = -\frac{\delta}{\delta\beta_s} S^*(\beta) \\
&= \sum_{i=1}^{n} \Bigg\{ y_i^* \left[-\frac{\delta}{\delta\beta_s} \Bigg(\frac{x_i \cdot (p_{11} - p_{01}) \cdot \exp(\beta^t x_i)}{\Big(\exp(\beta^t x_i) + 1 \Big) \Big(p_{11} \cdot \exp(\beta^t x_i) + p_{01} \Big)} \Bigg) \right] + \\
&\quad (1 - y_i^*) \left[-\frac{\delta}{\delta\beta_s} \Bigg(\frac{x_i \cdot (p_{10} - p_{00}) \cdot \exp(\beta^t x_i)}{\Big(\exp(\beta^t x_i) + 1 \Big) \Big(p_{10} \cdot \exp(\beta^t x_i) + p_{00} \Big)} \Bigg) \right] \Bigg\} \\
&= \sum_{i=1}^{n} \Bigg\{ y_i^* \left[-\frac{x_{ij} x_{is} \cdot (p_{11} - p_{01}) \cdot \exp(\beta^t x_i) \Big(p_{11} \cdot \exp(2\beta^t x_i) - p_{01} \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(p_{11} \cdot \exp(\beta^t x_i) + p_{01} \Big)^2} \right] + \\
&\quad (1 - y_i^*) \left[-\frac{x_{ij} x_{is} \cdot (p_{10} - p_{00}) \cdot \exp(\beta^t x_i) \Big(p_{10} \cdot \exp(2\beta^t x_i) - p_{00} \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(p_{10} \cdot \exp(\beta^t x_i) + p_{00} \Big)^2} \right] \Bigg\}
\end{aligned} \tag{29}
$$

Analog wie in Unterkapitel 5.2.1 (Logistische Regression ohne Anonymisierung) kann der Newton-Raphson-Algorithmus mittels der Ableitungen (28) sowie (29) so lange iteriert werden, bis die Schätzung für β konvergiert. Eine gute Wahl für den initialen Wert $\beta^{(0)}$ ist das geschätzte $\hat{\beta}$ der Logistischen Regression appliziert auf den perturbierten Daten (van den Hout, 1999, S. 63).

Rechenweg der Ableitung der Score Funktion

$$S^*(\beta) = \frac{\delta}{\delta\beta_j} l^*(\beta) = \sum_{i=1}^{n} \Bigg\{ y_i^* \left[\frac{\delta}{\delta\beta_j} \log\Bigg(p_{01} + (p_{11} - p_{01}) \Big(\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \Big) \Bigg) \right] + (1 - y_i^*) \left[\frac{\delta}{\delta\beta_j} \log\Bigg(p_{00} + (p_{10} - p_{00}) \Big(\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \Big) \Bigg) \right] \Bigg\}$$

Der Einfachheit halber wird zunächst der erste Teil betrachtet und abgeleitet. Somit:

$$\begin{aligned}
\frac{\delta}{\delta\beta_j}(1.) &= \frac{\delta}{\delta\beta_j} \log\Bigg(p_{01} + (p_{11} - p_{01}) \Big(\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \Big) \Bigg) \\
&= \frac{1}{\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01}} \cdot \frac{\delta}{\delta\beta_j} \left[\frac{(p_{11} - p_{01}) \cdot \exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} + p_{01} \right] \\
&= \frac{(p_{11} - p_{01}) \frac{\delta}{\delta\beta_j} \left[\frac{\exp(\beta^t x_i)}{1 + \exp(\beta^t x_i)} \right] + \frac{\delta}{\delta\beta_j} \left[p_{01} \right]}{\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01}} \\
&= \frac{(p_{11} - p_{01}) \cdot \frac{\frac{\delta}{\delta\beta_j} \left[\exp(\beta^t x_i) \right] \cdot \left(\exp(\beta^t x_i) + 1 \right) - \exp(\beta^t x_i) \cdot \frac{\delta}{\delta\beta_j} \left[\exp(\beta^t x_i) + 1 \right]}{\left(\exp(\beta^t x_i) + 1 \right)^2} + 0}{\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01}} \\
&= \frac{(p_{11} - p_{01}) \Big(\exp(\beta^t x_i) \frac{\delta}{\delta\beta_j} [\exp(\beta^t x_i)] (\exp(\beta^t x_i) + 1) - \exp(\beta^t x_i) \Big(\frac{\delta}{\delta\beta_j} [\exp(\beta^t x_i)] + \frac{\delta}{\delta\beta_j} [1] \Big) \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01} \Big)} \\
&= \frac{(p_{11} - p_{01}) \Big(\exp(\beta^t x_i) x_{ij} \frac{\delta}{\delta\beta_j} [\exp(\beta^t)] (\exp(\beta^t x_i) + 1) - \exp(\beta^t x_i) \Big(\exp(\beta^t x_i) \frac{\delta}{\delta\beta_j} [\exp(\beta^t x_i)] \Big) \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01} \Big)} \\
&= \frac{(p_{11} - p_{01}) \Big(\exp(\beta^t x_i) \cdot x_{ij} \cdot \Big(\exp(\beta^t x_i) + 1 \Big) - x_{ij} \cdot \frac{\delta}{\delta\beta_j} \big[\beta \big] \cdot \exp(2\beta^t x_i) \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01} \Big)} \\
&= \frac{(p_{11} - p_{01}) \Big(x_{ij} \cdot \exp(\beta^t x_i) \cdot \Big(\exp(\beta^t x_i) + 1 \Big) - x_{ij} \cdot \exp(2\beta^t x_i) \Big)}{\Big(\exp(\beta^t x_i) + 1 \Big)^2 \Big(\frac{(p_{11} - p_{01}) \exp(\beta^t x_i)}{\exp(\beta^t x_i) + 1} + p_{01} \Big)} \\
&= \frac{x_{ij} \cdot (p_{11} - p_{01}) \exp(\beta^t x_i)}{\Big(\exp(\beta^t x_i) + 1 \Big) \Big(p_{11} \exp(\beta^t x_i) + p_{01} \Big)}
\end{aligned}$$

Das Analogon erfolgt beim zweiten Teil und das gesamte Ergebnis der beiden ergibt

das Resultat in (28). Auf den Rechenweg der Ableitung der Fisher Information Funktion wird verzichtet, da er sehr umständlich wird und mit vielen Zeilenumbrüchen verbunden ist, was die Nachvollziehbarkeit erschwert.

6.2.2 (β)-Korrekturansatz für X^* mittels EM-Algorithmus

Dieser Teil beschäftigt sich mit einem Korrekturansatz einer Kovariable (anstelle einer Zielvariable wie in 6.2.1), welche mittels PRAM kontaminiert wurde (siehe 5.2.3). Analog zum Korrekturansatz für die Zielvariable in 6.2.1 wird auch hier angenommen, dass die Transitions- bzw. Markov Matrix mit den perturbierten Daten von den NSIs veröffentlicht wird.

van den Hout (1999, S. 64-66) schlägt einen Korrekturansatz mittels EM-Algorithmus in einer Logistischen Regression vor, wenn eine Zielvariable durch PRAM kontaminiert wird. Wenn nun eine Kovariable mittels PRAM perturbiert wird, kann der EM-Algorithmus ebenso verwendet werden, um Schätzungen für β zu erhalten. Da nun die log Likelihood nicht linear in X ist (wie es bei der perturbierten Zielvariable der Fall ist), hat der EM-Algorithmus (in 5.2.4 beschrieben) van den Hout (1999) zufolge keine einfache Form. Dafür wird dennoch der bedingte Erwartungswert der log Likelihood (d.h. $Q(\beta, \beta^{(q)})$) beim E-Step anhand Monte Carlo Simulation geschätzt und beim M-Step seine Schätzung maximiert. In diesem Fall wird der EM-Algorithmus Monte Carlo EM (MCEM) genannt, dessen Anwendung auch in allgemeinen Situationen erfolgt, bei denen der E-Step ebenso komplex zu schätzen ist (van den Hout, 1999, S. 67).

Im vorliegenden Fall, mit X^* in einer Logistischen Regression, kann der bedingte Erwartungswert beim E-Step wie folgt bestimmt werden (idem):

$$\begin{aligned} Q(\beta, \beta^{(q)}) &= \mathbb{E}_X\left[\log(\mathcal{L}(\beta)|y, X)|X^*, \beta^{(q)}\right] \\ &= \mathbb{E}_X\Big[\sum_{i=1}^{n}\Big(y_i \log(\pi(X_i)) + (1-y_i)\log(1-\pi(X_i))\Big)|X^*, \beta^{(q)}\Big] \end{aligned} \tag{30}$$

Aus (30) lässt sich feststellen, dass die bedingte Verteilung von X_i gegeben X_i^*, Y_i und $\beta^{(q)}$ für die Monte Carlo Simulation im E-Step benötigt wird, deren Definition auf van den Hout (1999, S. 67-68) beruht:

$$
\begin{aligned}
P(X_i = k|X_i^* = l, Y_i = h, \beta^{(q)}) &= \frac{P(Y_i = h, X_i = k, X_i^* = l|\beta^{(q)})}{P(Y_i = h, X_i^* = l|\beta^{(q)})} \\
&= \frac{P(Y_i = h|X_i = k, X_i^* = l, \beta^{(q)})P(X_i = k, X_i^* = l)}{P(Y_i = h, X_i = 1, X_i^* = l|\beta^{(q)}) + P(Y_i = h, X_i = 0, X_i^* = l|\beta^{(q)})} \\
&= \frac{P(Y_i = h|X_i = k, \beta^{(q)})P(X_i^* = l|X_i = k)P(X_i = k)}{\sum_{k=0}^{1} P(Y_i = h|X_i = k, X_i^* = l, \beta^{(q)})P(X_i = k, X_i^* = l)} \\
&= \frac{P(Y_i = h|X_i = k, \beta^{(q)})P(X_i^* = l|X_i = k)P(X_i = k)}{\sum_{k=0}^{1} P(Y_i = h|X_i = k, \beta^{(q)})P(X_i^* = l|X_i = k)P(X_i = k)} \\
&= \frac{p_{kl}P(Y_i = h|X_i = k, \beta^{(q)})P(X_i = k)}{\sum_{k=0}^{1} p_{kl}P(Y_i = h|X_i = k, \beta^{(q)})P(X_i = k)}
\end{aligned}
\tag{31}
$$

In den zweiten und dritten Schritten (beim Zähler bzw. beim Nenner) der letzten Gleichung (31) ersetzt $P(Y_i = h|X_i = k)$ die $P(Y_i = h|X_i = k, X_i^* = l)$, da X_i^* hinsichtlich Y_i irrelevant wird, wenn $X_i = k$ gegeben ist. Ferner wurde $P(X_i^* = l|X_i = k)$ durch p_{kl} ersetzt, was anhand der Markov Matrix gegeben ist und hier (2) in der Beschreibung von PRAM definiert wurde.

Darüber hinaus ist $P(X_i = k)$ in der letzten Gleichung (31) nicht bekannt. Die Schätzung davon wird dennoch mittels des Moment Schätzers von van den Hout (1999, S. 68) vorgeschlagen. Durch seine Anwendung können die originalen Häufigkeiten der Variable X wiederum wie folgt bestimmt werden:

$$\hat{T}_{X_i} = (\boldsymbol{P}_X)^{-1} T_{X_i^*} \tag{32}$$

mit $T_{X_i^*} = \left(T_{X_i^*}(0), T_{X_i^*}(1)\right)$ als Vektor der beobachteten Häufigkeiten der perturbierten Variable X^*, $\hat{T}_{X_i} = \left(\hat{T}_{X_i}(0), \hat{T}_{X_i}(1)\right)$ als entsprechender Vektor der geschätzten Häufigkeiten der originalen Variable X und $\boldsymbol{P}_X$ als angewendete Transitionsmatrix mittels PRAM für die Anonymisierung der ursprünglichen Variable X. Daraus erfolgt (idem):

$$\hat{P}(X_i = k) = \frac{\hat{T}_{X_i}(k)}{n}$$

wobei $\hat{P}(X_i = k)$ die geschätzte Wahrscheinlichkeit der Variable X für Kategorie k für Beobachtung i und n die Anzahl der gesamten Beobachtungen sind.

Anschließend findet der Monte Carlo E-Step wie folgt statt. In jeder q-ten Iteration wird $x_i^1, ..., x_i^M$ (mit M die Anzahl der Monte Carlo Wiederholungen und $x_i^m = (x_{i0}^m, x_{i1}^m)$) aus der Verteilung von X_i gegeben $x_i^* = (x_{i0}^*, x_{i1}^*), y_i$ und $\beta^{(q)}$ gezogen. Danach wird die Q-Funktion (30) approximiert (idem):

$$\hat{Q}(\beta, \beta^{(q)}) : \beta \rightarrow \frac{1}{M} \sum_{m=1}^{M} \log\Big(\mathcal{L}(\beta^{(q)})|y, X^m\Big) \tag{33}$$

Dann erfolgt der M-Step, welcher $\hat{Q}(\beta, \beta^{(q)})$ über β maximiert, um $\beta^{(q+1)}$ zu erhalten. Hier kommt die normale Logistische Regression zur Anwendung, da $\log\Big(\mathcal{L}(\beta^{(q)})|y, X^m\Big)$ in 33 van den Hout (1999) zufolge eigentlich schon eine Aufsummierung ist, bei der "we need to maximize a complete-data log likelihood which is made up of a data set which is M times larger than the original data set"(van den Hout, 1999, S. 68), wie folgt:

$$\max_{\beta} \sum_{m=1}^{M} \sum_{i=1}^{n} \Big(y_i \log(\pi(X_i^m)) + (1 - y_i)\log(1 - \pi(X_i^m))\Big) \tag{34}$$

Analog wie beim Newton-Raphson-Algorithmus (6.2.1) stellt das geschätzte $\hat{\beta}$ der auf den perturbierten Daten angewendeten Logistischen Regression van den Hout (1999) zufolge auch hier einen guten Start für $\beta^{(0)}$ dar.

Noch im Rahmen des MCEM-Algorithmus betont van den Hout (1999, S. 68), dass die Wahl von M und die Überwachung der Konvergenz hinsichtlich β Schwierigkeiten mit sich bringt. Vorgeschlagen ist dennoch: 1) mit einem kleinen Wert M zu starten und ihn in der Konvergenzzone zu erhöhen; 2) hinsichtlich der Überwachung der Konvergenz $\beta^{(q)}$ gegenüber q zu tabellieren, bis Erstgenanntes sich stabilisiert (van den Hout, 1999, S. 68-69).

6.2.3 Korrekturansatz für individuelle Prädiktionen y_i bei binärer X^*

Da kein Ansatz für eine Korrektur der individuellen Prädiktionen (d.h. y_i) in der Literatursuche gefunden wurde und dies dennoch für sehr wichtig gehalten wird, wird hier eine eigene Überlegung diesbezüglich entwickelt, in `R` implementiert und abschließend getestet bzw. evaluiert. Die Ergebnisse dieser Simulation sind in Tabelle 27 am Schluss zu finden.

Für die Darstellung der Idee wird folgende Situation angenommen: Es sei T_{X^*} die beobachteten Häufigkeiten einer binären perturbierten Variable X^* mit Kategorien 0 und 1 und insgesamt 180 Beobachtungen

$$T_{X^*} = \Big(T_{X^*}(0) = 80, T_{X^*}(1) = 100\Big) = 180$$

und folgende mitgelieferte Transitionsmatrix der entsprechenden Variable:

$$\boldsymbol{P}_X = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}$$

Daraus folgt, dass von den 80 beobachteten perturbierten Nullen 64 ($= 0.8 \cdot 80$) richtig (im Durchschnitt) und 16 ($= 0.2 \cdot 80$) inkorrekt (auch im Durchschnitt) zu erwarten sind. Analog sind bei den Einsen 80 ($= 0.8 \cdot 100$) richtig und 20 ($= 0.2 \cdot 100$) inkorrekt (beide ebenso im Durchschnitt betrachtet) zu erwarten. Das heißt, 16 Nullen und 20 Einsen müssten (im Durchschnitt) korrigiert werden, um die wahre Prädiktion zu erhalten. Ferner würde Sinn ergeben, anzunehmen, dass bei den wahren Werten die Accuracy (in 4.1 bereits dargestellt) höher als bei den perturbierten ist, denn das ist letztendlich die wahre Beziehung zwischen X und Y.

Auf dieser Basis könnten folgende Schritte erfolgen, um eine Korrektur der individuellen Prädiktionen y_i vorzunehmen:

1. Fitte eine Logistische Regression auf die gelieferten perturbierten Daten und speichere die Accuracy oder eine andere Performance Metrik ab, um diese Performance mit den folgenden vergleichen zu können (Benchmark-Rolle).

2. Dann betrachte zunächst alle 0-Werte der perturbierten Variable und tausche bei der ersten Beobachtung die 0 auf 1 um. Mit der Konstanthaltung aller anderen Werte kann diese individuelle Veränderung untersucht bzw. experimentiert werden, was passiert, wenn sie eigentlich den Gegenwert hätte (Was-wenn-Rolle).

3. Fitte anschließend erneut eine Logistische Regression auf die gesamten perturbierten Daten, mit der einzigen im vorherigen Schritt vorgenommenen Veränderung, und speichere ebenfalls die Performance Metrik ab. Somit kann mittels Performance Metrik (Performance-Metrik-Vergleich-Rolle) der Effekt der individuellen Veränderung dieser Beobachtung analysiert bzw. geschätzt werden.

4. Mache zum einen die Veränderung im zweiten Schritt rückgängig und betrachte zum anderen nun die zweite 0 und tausche sie auf 1 um. Somit kann die zweite individuelle Beobachtung bzw. deren Was-wenn-Rolle betrachtet werden.

5. Führe die Prozedur analog für alle 0-Werte durch, um für alle Beobachtungen einen Veränderungseffekt untersuchen zu können.

6. Da 16 Beobachtungen mit inkorrektem 0-Wert zu erwarten sind, selektiere aus dem durchgeführten Prozess die 16 Beobachtungen, welche die Accuracy am meisten erhöht haben, denn in diesem Fall könnte der Schluss gezogen werden, sie seien nun korrigiert worden, da die Performance Metrik “optimiert” wurde.

7. Führe die gleiche Prozedur bei den 1-Werten entsprechend durch, um eine bessere Performance bzw. eine darauffolgende Korrektur bei denen analog zu erhalten.

8. Schließlich kann eine finale korrigierte Logistische Regression durchgeführt werden, unter Berücksichtigung aller vorgenommenen Korrekturen, und die Accuracy kann mit der des Ausgangsmodells gegenübergestellt werden. Es wird eine Verbesserung hinsichtlich des korrigierten Modells bei der Accuracy erwartet.

Dieser gerade entwickelte Algorithmus wurde in `R` mittels einer selbstgeschriebenen Funktion in einer weiteren Simulation implementiert und getestet. Dabei wurde der Seed 102022, eine Transitionsmatrix wie in 6.2.3 definiert und verschiedene Häufigkeiten in den zwei Kategorien, die X bzw. X^* annehmen, untersucht, nämlich als $T_{X^*}(0)$ (für die Klasse 0) und $T_{X^*}(1)$ (für die Klasse 1) bezeichnet. Als Performance Metrik wurde die Accuracy herangezogen. Die Ergebnisse dazu sind in der Tabelle 27 präsentiert:

mittels PRAM perturbierter X^*	Accuracy Benchmark (ohne Korrektur)	Accuracy Korrektur	Accuracy Benchmark Standard-abweichung	Accuracy Korrektur Standard-abweichung
$\mathrm{T}_{X^*}(0) = 10, T_{X^*}(1) = 20$	0,5962	0,8095	0,0967	0,1293
$\mathrm{T}_{X^*}(0) = 100, T_{X^*}(1) = 200$	0,4767	0,7000	0,0560	0,1728
$\mathrm{T}_{X^*}(0) = 1.000, T_{X^*}(1) = 2.000$	0,4867	0,7052	0,0138	0,2615
$\mathrm{T}_{X^*}(0) = 20, T_{X^*}(1) = 10$	0,6029	0,8095	0,1861	0,2110
$\mathrm{T}_{X^*}(0) = 200, T_{X^*}(1) = 100$	0,5333	0,7333	0,0565	0,2306
$\mathrm{T}_{X^*}(0) = 2.000, T_{X^*}(1) = 1.000$	0,4930	0,6907	0,0148	0,2749

Tabelle 27: Vergleich der Accuracy ohne vs. mit der entwickelten Korrektur der individuellen Prädiktionen bei binärer perturbierter Variable

Aus der obigen Tabelle lässt sich feststellen, dass die vorgeschlagene Korrektur zu einer wesentlichen Erhöhung der Accuracy (von circa 30%) bei den Prädiktionen führt, was sie somit anwendbar bzw. fruchtbar macht.

6.2.4 Korrekturansatz für individuelle Prädiktionen y_i bei binärer Y^*

Da bei dem vorherigen Teil die Korrektur zu einer deutlichen Verbesserung der Accuracy geführt hat, kann hier eine logische Frage auftreten: Könnte man diese Vorteile nicht auch auf den Fall übertragen, bei dem anstelle einer Kovariable eine binäre Zielvariable perturbiert wurde? Das Analogon aus 6.2.3 könnte ebenso für Y^* angewendet werden, um die individuellen Prädiktionen y_i^* korrigieren zu können. Mit dieser Untersuchung beschäftigt sich dieser Teil.

Wie bei 6.2.3 werden auch hier eine ähnliche Transitionsmatrix angenommen

$$\boldsymbol{P}_Y = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}$$

und die hier 6.2.3 entsprechenden formulierten Schritte vorgenommen (nun bezüglich der Zielvariable Y^* anstelle der Kovariable X^*).

Die ganze Prozedur wurde ebenso analog zu der vorherigen implementiert, um auch die daraus folgenden Effekte zu erkennen. Die Resultate sind in Tabelle 28 veranschaulicht:

mittels PRAM perturbierter Y^*	Accuracy Benchmark (ohne Korrektur)	Accuracy Korrektur	Accuracy Benchmark Standard-abweichung	Accuracy Korrektur Standard-abweichung
$\mathrm{T}_{X^*}(0) = 10, T_{X^*}(1) = 20$	0,6343	0,5676	0,0455	0,0707
$\mathrm{T}_{X^*}(0) = 100, T_{X^*}(1) = 200$	0,5933	0,5533	0,0071	0,0046
$\mathrm{T}_{X^*}(0) = 1.000, T_{X^*}(1) = 2.000$	0,5920	0,5553	0,0005	0,0005
$\mathrm{T}_{X^*}(0) = 20, T_{X^*}(1) = 10$	0,6333	0,5638	0,0745	0,1990
$\mathrm{T}_{X^*}(0) = 200, T_{X^*}(1) = 100$	0,5300	0,5467	0,0591	0,0046
$\mathrm{T}_{X^*}(0) = 2.000, T_{X^*}(1) = 1.000$	0,6107	0,5667	0,0009	0

Tabelle 28: Vergleich der Accuracy ohne vs. mit der entwickelten Korrektur der individuellen Prädiktionen bei binärer perturbierter Zielvariable

Aus dieser Tabelle lässt sich konstatieren, dass der Versuch der Anwendung der Korrekturmethode auf die perturbierte Zielvariable eigentlich nichts gebracht, sondern eher zu einer Verschlechterung gegenüber dem Benchmark Modell geführt hat. Eine andere daraus resultierende Überlegung bestand darin, nur die $T_{X^*}(0)$ zu korrigieren. In manchen Situationen, wie z.B. Krebserkennung, mag der Fokus stärker auf dem richtigen prädiktierten $Y = 1$ liegen. Auf Grund dessen kann die Korrektur der "falsch" klassifizierten $Y^* = 0$ zu korrektem $Y^* = 1$ von großer Bedeutung sein. Darauf basierend wurde diese Überlegung erneut in `R` untersucht, was zu folgenden Ergebnissen geführt hat (siehe Tabelle 29):

mittels PRAM perturbierter Y^*	Accuracy Benchmark (ohne Korrektur)	Accuracy Korrektur	Accuracy Benchmark Standard-abweichung	Accuracy Korrektur Standard-abweichung
$\mathrm{T}_{X^*}(0) = 10, T_{X^*}(1) = 20$	0,6343	0,7029	0,0455	0,0581
$\mathrm{T}_{X^*}(0) = 100, T_{X^*}(1) = 200$	0,5933	0,6734	0,0071	0,0073
$\mathrm{T}_{X^*}(0) = 1.000, T_{X^*}(1) = 2.000$	0,5920	0,6737	0,0005	0,0005
$\mathrm{T}_{X^*}(0) = 20, T_{X^*}(1) = 10$	0,6333	0,4333	0,0745	0,1900
$\mathrm{T}_{X^*}(0) = 200, T_{X^*}(1) = 100$	0,5300	0,5033	0,0591	0,0502
$\mathrm{T}_{X^*}(0) = 2.000, T_{X^*}(1) = 1.000$	0,6107	0,5057	0,0009	0,0095

Tabelle 29: Vergleich der Accuracy ohne vs. mit der entwickelten (teilweise) Korrektur der individuellen Prädiktionen bei binärer perturbierter Zielvariable; Anwendung nur auf $T_{X^*}(0)$

Daraus ist erkennbar, dass die Anwendung der Korrektur nur Vorteile im Falle $T_{X^*}(1) \gg T_{X^*}(0)$ bringt, d.h. falls die Anzahl bzw. Häufigkeiten in der Klasse 1 überwiegend

bzw. deutlich höher als in der Klasse 0 ist. Andernfalls verringert sich die Accuracy. Auf Grund dessen war die Korrektur, angewendet auf der binären perturbierten Zielvariable, nicht erfolgreich. Eine künftige Forschung in dieser Richtung mag somit hilfreich sein, um die individuellen Prädiktionen zu korrigieren.

7 Konklusion und Ausblick

Der Schutz der Privatsphäre sowie die "consequences of the concept 'big brother is watching you'" (Domingo-Ferrer et al., 2012, S. xi) sind zu einem immer wichtigeren Anliegen der Gesellschaft geworden. Die SDC-Methoden kommen in diesem Zusammenhang zur Anwendung, indem sie versuchen, einen Trade-off zwischen Offenlegungsrisiko und Nutzen der Daten bzw. Informationsverlust zu erreichen. Mit Blick auf die große Bedeutung dieses Themas hat sich diese Masterthesis mit der Anwendung verschiedener ML-Ansätze auf mit diversen Anonymisierungsverfahren perturbierte Daten auf Mikroebene beschäftigt. Des Weiteren wurde untersucht, wie sensibel ML-Algorithmen auf die Anonymisierung der Daten reagieren, welche Effekte das (absichtliche oder nicht) Kontaminieren der Daten verursacht, wie aus diesen perturbierten Daten dennoch Informationen gewonnen werden können, sowie auch wie sie adjustiert bzw. korrigiert werden können, um valide Inferenz zu betreiben.

Was das Offenlegungsrisiko und den Informationsverlust anbelangt, haben sich die Behauptungen von Benschop and Welch (2019) in 3.2.1 mithilfe einer Simulation widerspiegeln lassen, indem die beiden folgenden Punkte den Informationsverlust verringern: 1) zum einen die simultane Perturbation ausschließlich innerhalb korrelierter Gruppen bzw. Variablen anstelle von allen, sowie 2) zum anderen je stärker die Variablen korrelieren. Letztgenanntes trifft nur bei der Microaggregation zu, während Erstgenanntes neben der Microaggregation auch bei *additive* und *correlated2* Noise zu sehen ist.

Als finales Ergebnis der angewendeten ML-Ansätze auf den anonymisierten Daten mit metrischer Zielvariable konnte Folgendes festgestellt werden: Local Suppression für eine weitere Reduzierung des Offenlegungsrisikos wurde in der gesamten Arbeit komplett ausgeschlossen, da ML-Algorithmen sehr sensibel auf fehlende Werte reagieren. Somit kommen die Beobachtungen mit fehlenden Werten gar nicht zum Einsatz. Ferner sind Neuronale Netze robust gegen Anonymisierung, was bedeutet, dass der RMSE der originalen und der perturbierten Daten unverändert bleibt, unabhängig von der Menge an Perturbation sowie von Verfahren. Des Weiteren, ist die Variable unwichtig im Sinne von Variable Importance, unabhängig von starker oder schwacher Korrelationsstruktur, gibt es kaum, wenn überhaupt, einen Unterschied beim RMSE nach der Perturbation (sowohl

bei additiver unkorrelierter als auch bei korrelierter Noise). Ferner, je wichtiger eine unkorrelierte mit Noise Addition zu perturbierende nicht normalverteilte Variable wird, desto deutlicher steigt der RMSE. Ebenso höher wird der RMSE, wenn die Variable Importance steigt, während die nicht normalverteilte Variable gleichzeitig korreliert ist. Zudem: Je stärker sie korreliert wird und gleichzeitig sehr wichtig ist, desto höher wird der RMSE. Allerdings gelten die letzten drei Aussagen nur, wenn die Variable nicht normalverteilt ist; wenn sie aus einer Normalverteilung stammt, sind keine erhebliche Steigerungen des RMSEs weder bei additiver noch bei *correlated2* Noise Addition zu erwarten. Zudem ist auch zu erkennen, dass bei perturbierten Daten mit Microaggregation (mit den diversen aggregierten ks) kein großer Unterschied hinsichtlich des RMSEs erkennbar war und dass es zum anderen sowohl bei Gradient Boosting Machine als auch bei Random Forest keinen großen Unterschied hinsichtlich der Variable Importance gab. Anders ist es allerdings bei sowohl *additive* als auch *correlated2* Noise Addition. In diesem Fall verlieren die perturbierten Variablen sehr schnell an Bedeutung. Am drastischsten ist dies beim additiven Fall.

Anschließend wurde in 6.1.5 eine Korrektur für β mittels SIMEX Methode bei einem Linearen Regressionsmodell und additiven Messfehler auf eine metrische Kovariable präsentiert, in `R` angewendet und evaluiert. Daraus lässt sich feststellen, dass anhand dieser Methode eine konsistente Schätzung des Koeffizienten für eine metrische perturbierte Kovariable erhalten werden kann, wenn der addierte Messfehler nicht allzu groß ist. Je höher Letztgenannter wird, umso verzerrter wird die Schätzung. Allerdings ist diese immer noch wesentlich genauer als jene aus dem naiven Modell. Ferner wurde in 6.1.6 eine Korrektur für die individuellen Prädiktionen y_i mittels Regressionskalibrierung diskutiert, welche nicht erfolgreich war.

Darüber hinaus wurden in 6.2.1 und 6.2.2 Korrekturansätze hinsichtlich β mittels Newton-Raphson- und EM-Algorithmus im Rahmen der Anwendung von PRAM auf eine binäre Zielvariable sowie eine binäre Kovariable in einer Logistischen Regression in Betracht gezogen. Dabei wurden die Likelihood Funktionen in geeigneter Form adjustiert, um die Perturbation zu berücksichtigen. Ein anderes Korrekturverfahren hinsichtlich β ist mittels SIMEX bzw. MisClassification SIMEX (MCSIMEX) Methode, deren Details aus Lederer and Küchenhoff (2006) und Küchenhoff et al. (2006) entnommen werden können. Ferner wurden zwei weitere Korrekturmethoden entwickelt und ausprobiert, welche die individuellen Prädiktionen y_i (versuchen zu) adjustieren, wenn zum einen eine binäre Kovariable X^* (6.2.3) sowie zum anderen eine binäre Zielvariable Y^* (6.2.4) perturbiert wurden, um eine höhere Accuracy zu erreichen. Nur der erste Korrekturansatz (mit X^*) hat sich dabei als fruchtbar erwiesen.

Was noch die Anwendung von PRAM auf kategoriale Variable anbelangt, schlagen

van den Hout and van der Heijden (2002, S. 273-278) Korrekturverfahren auch in Bezug auf die Kontingenztabellen mittels Moment- sowie Maximum Likelihood Schätzer und deren entsprechende Kovarianzen vor. Die selben Autoren befassen sich ebenfalls mit der Korrektur von Odds Ratios, auch mittels der beiden gerade erwähnten Schätzer (siehe van den Hout and van der Heijden (2002, S. 281-283)).

In dieser Thesis wurde nur ein Anonymisierungsverfahren simultan verwendet. Eine andere interessante Frage, die an dieser Stelle auftreten kann, ist: Wie stark würden mehrere gleichzeitig angewendete Anonymisierungstools die Effekte und Analysen beeinflussen? Diese Frage könnte ein neues Forschungsfenster öffnen. Darüber hinaus wäre eine weitere logische Konsequenz die Untersuchung, welche Auswirkung die Kontaminierung einer metrischen Zielvariable hat, was in manchen realen Situationen der Fall ist. Ferner könnte analysiert werden, was mit den Daten passiert, wenn sowohl die Zielvariable als auch eine oder mehrere Kovariablen perturbiert werden (im metrischen und diskreten Fall), und wie die Analysen in diesem Zusammenhang adjustiert werden sollten. van den Hout (1999, S. 69) behauptet, der MCEM-Algorithmus, der auch in 6.2.2 zur Anwendung kam, könnte herangezogen werden, wenn sowohl die binäre Outcome Variable als auch eine binäre Kovariable gleichzeitig durch PRAM perturbiert werden, um eine Korrektur der Messfehler vorzunehmen. Eine weitere zukünftige Aufgabenstellung könnte sein, eine Korrektur der individuellen y_i bei einer binären perturbierten Zielvariable zu entwickeln, was in 6.2.4 versucht wurde, leider aber nicht erfolgreich war.

Dem Datenschutz mit seinen vielfältigen Bestimmungen und Auswirkungen kommt in der modernen Gesellschaft von heute eine immer größere Bedeutung zu. Deshalb könnte diese Thematik auch für die Universitäten und deren Studenten künftig von noch größerem Interesse sein, z.B. in Form von Kursen bzw. Seminaren. Denn es gibt auf diesem Gebiet noch sehr viel mehr zu tun...

Literatur

Alfons, A. and Templ, M. (2013). Estimation of social exclusion indicators from complex surveys: The R package laeken, *Journal of Statistical Software* **54**(15): 1–25.

Benschop, T. and Welch, M. (2019). Statistical disclosure control for microdata: A practice guide. Unter: `https://sdcpractice.readthedocs.io/en/latest/` (abgerufen am 06.11.2022).

Bonaccorso, G. (2018). *Mastering Machine Learning Algorithms: Expert Techniques to Implement Popular Machine Learning Algorithms and Fine-Tune Your Models*, Packt Publishing.

Brand, R. and Giessing, S. (2002). Report on preparation of the data set and improvements on sullivans algorithm. Unter: `https://research.cbs.nl/casc/deliv/11d1.pdf` (abgerufen am 06.11.2022).

Breiman, L. (2001). Random forests, *Machine Learning* **45**: 5–32.

Caiola, G. and Reiter, J. P. (2010). Random forests for generating partially synthetic, categorical data, *Transactions on Data Privacy* **3**(1): 27–42.

Cano, I. and Torra, V. (2011). Edit constraints on microaggregation and additive noise, *in* C. Dimitrakakis, A. Gkoulalas-Divanis, A. Mitrokotsa, V. S. Verykios and Y. Saygin (eds), *Privacy and Security Issues in Data Mining and Machine Learning: International ECML/PKDD Workshop, PSDML 2010, Barcelona, Spain, September 24, 2010. Revised Selected Papers*, Springer-Verlag Berlin Heidelberg 2011, p. 1–14.

Carroll, R. J., Ruppert, D., Stefanski, L. A. and Crainiceanu, C. M. (2006). *Measurement Error in Nonlinear Models: A Modern Perspective*, 2 edn, Chapman and Hall/CRC.

Cook, J. and Stefanski, L. (1994). Simulation-extrapolation estimation in parametric measurement error models, *Journal of the American Statistical Association* **89**(428): 1314–1328.

Cook, R. D. and Weisberg, S. (1982). Criticism and influence analysis in regression, *Sociological Methodology* **13**: 313–361. Unter: `https://doi.org/10.2307/270724` (abgerufen am 06.11.2022).

Dandekar, R. A., Domingo-Ferrer, J. and Sebé, F. (2002). LHS-based hybrid microdata vs rank swapping and microaggregation for numeric microdata protection, *in* J. Domingo-Ferrer (ed.), *Inference Control in Statistical Databases: from theory to practice, LNCS 2316*, Springer-Verlag Berlin Heidelberg 2002, p. 153–162.

de Wolf, P.-P. (2006). Risk, utility and PRAM, *in* J. Domingo-Ferrer and L. Franconi (eds), *Privacy in Statistical Databases: CENEX-SDC Project International Conference, PSD 2006 Rome, Italy, December 13-15, 2006 Proceedings, LNCS 4302*, Springer-Verlag Berlin Heidelberg 2006, p. 189–204.

de Wolf, P.-P., Gouweleeuw, J. M., Kooiman, P. and Willenborg, L. (2002). Reflections on PRAM. Unter: `https://research.cbs.nl/casc/Related/Sdp_98_2.pdf` (abgerufen am 06.11.2022).

de Wolf, P.-P. and van Gelder, I. (2004). An empirical evaluation of PRAM. Unter: `https://research.cbs.nl/casc/Related/discussion-paper-04012.pdf` (abgerufen am 06.11.2022).

Defays, D. and Anwar, M. (1998). Masking microdata using micro-aggregation, *Journal of Official Statistics* **14**(4): 449–461.

Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E., Spicer, K., de Wolf, P. and Hundepool, A. (2012). *Statistical Disclosure Control*, Wiley Series in Survey Methodology, Wiley.

Domingo-Ferrer, J., Mateo-Sanz, J., Oganian, A. and Torres, A. (2002). On the security of microaggregation with individual ranking: Analytical attacks, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **10**(5): 477–492.

Domingo-Ferrer, J., Mateo-Sanz, J. and Torra, V. (2001). Comparing SDC methods for microdata on the basis of information loss and disclosure risk. Unter: `https://crises-deim.urv.cat/web/docs/publications/conferences/597.pdf` (abgerufen am 06.11.2022).

Fahrmeir, L., Hamerle, A. and Tutz, G. (eds) (2015). *Multivariate statistische Verfahren*, De Gruyter, Berlin, Boston. Unter: `https://doi.org/10.1515/9783110816020` (abgerufen am 06.11.2022).

Gouweleeuw, J., Kooiman, P., Willenborg, L. and De Wolf, P. (1998). Post randomisation for statistical disclosure control: Theory and implementation, *Journal of Official Statistics* **14**(4): 463–478.

Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Lenz, R., Longhurst, J., Nordholt, E. S., Seri, G. and Wolf, P.-P. D. (2006). *Handbook on Statistical Disclosure Control*, Vol. Version 1.0, CENEX SDC: a CENtre of EXcellence for Statistical Disclosure Control. Unter: `https://ec.europa.eu/eurostat/cros/system/files/CENEX-SDC_handbook.pdf` (abgerufen am 06.11.2022).

Kim, J. J. (1986). A method for limiting disclosure in microdata based on random noise and transformation, *Proceedings of the Section on Survey research methods*, American Statistical Association, pp. 370–374. Unter: `http://www.asasrms.org/Proceedings/y1986f.html` (abgerufen am 06.11.2022).

Kubat, M. (2021). *An Introduction to Machine Learning*, 3 edn, Springer Cham.

Kuhn, M. (2019). *The caret Package*. Unter: `https://topepo.github.io/caret/index.html` (abgerufen am 06.11.2022).

Kuhn, M. (2022). *caret: Classification and Regression Training*. R package version 6.0. Unter: `https://CRAN.R-project.org/package=caret`.

Küchenhoff, H., Mwalili, S. and Lesaffre, E. (2006). A general method for dealing with misclassification in regression: the misclassification SIMEX, *Biometrics* **62**(1): 85–96. Unter: `https://doi.org/10.1111/j.1541-0420.2005.00396.x` (abgerufen am 06.11.2022).

Lederer, W. and Küchenhoff, H. (2006). A short introduction to the SIMEX and MC-SIMEX, *R News* **6**(4): 26–31. Unter: `https://cran.r-project.org/doc/Rnews/Rnews_2006-4.pdf` (abgerufen am 06.11.2022).

Mateo-Sanz, J. M., Sebé, F. and Domingo-Ferrer, J. (2004). Outlier protection in continuous microdata masking, *in* J. Domingo-Ferrer and V. Torra (eds), *Privacy in Statistical Databases: CASC Project Final Conference, PSD 2004 Barcelona, Catalonia, Spain, June 9-11, 2004, Proceedings, LNCS 3050*, Springer-Verlag Berlin Heidelberg 2004, p. 201–215.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. Unter: `https://www.R-project.org/` (abgerufen am 08.11.2022).

Sanz, I. P. (2019). *Machine Learning with R Quick Start Guide: A beginner´s guide to implementing machine learning techniques from scratch using R 3.5*, Packt Publishing.

Shaw, P. (2017). Understanding and tackling measurement error: SIMEX, *in* P. Shaw and R. Keogh (eds), *Understanding and tackling measurement error: A whistle stop tour of modern practical methods*, pp. 57–81. 2017, Vortragsfolien. Unter: `http://www.biometrische-gesellschaft.de/arbeitsgruppen/weiterbildung/education-for-statistics-in-practice.html` (abgerufen am 06.11.2022).

Shlomo, N., Tudor, C. and Groom, P. (2010). Data swapping for protecting census tables, *in* J. Domingo-Ferrer and E. Magkos (eds), *Privacy in Statistical Databases: UNESCO Chair in Data Privacy, International Conference, PSD 2010, Corfu, Greece, September 22-24, 2010, Proceedings, LNCS 6344*, Springer-Verlag Berlin Heidelberg 2010, p. 41–51.

Templ, M. (2006). Software development for SDC in R, *in* J. Domingo-Ferrer and L. Franconi (eds), *Privacy in Statistical Databases: CENEX-SDC Project International Conference, PSD 2006 Rome, Italy, December 13-15, 2006 Proceedings, LNCS 4302*, Springer-Verlag Berlin Heidelberg 2006, p. 347–359.

Templ, M. (2007). sdcmicro: A new flexible R-package for the generation of anonymised microdata: Design issues and new methods. Unter: `http://www2.uaem.mx/r-mirror/web/packages/sdcMicro/vignettes/sdcMicroPaper.pdf` (abgerufen am 06.11.2022).

Templ, M. (2017). *Statistical Disclosure Control for Microdata: Methods and Applications in R*, Springer.

Templ, M., Kowarik, A. and Meindl, B. (2015). Statistical disclosure control for microdata using the R package sdcMicro, *Journal of Statistical Software* **67**(4): 1–36.

Templ, M. and Meindl, B. (2008). Robustification of microdata masking methods and the comparison with existing methods, *in* J. Domingo-Ferrer and Y. Saygın (eds), *Privacy in Statistical Databases: UNESCO Chair in Data Privacy International Conference, PSD 2008, Istanbul, Turkey, September 24-26, 2008, Proceedings, LNCS 5262*, Springer Berlin Heidelberg, pp. 113–126.

van den Hout, A. (1999). The analysis of data perturbed by PRAM. Unter: `https://www.ucl.ac.uk/%7Eucakadl/TwAIO.pdf` (abgerufen am 06.11.2022).

van den Hout, A. and van der Heijden, P. G. (2002). Randomized response, statistical disclosure control and misclassificatio: a review, *International Statistical Review* **70**(2): 269–288.

Wallace, M. (2020). Analysis in an imperfect world, *Significance* **17**(1): 14–19.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York. Unter: `https://ggplot2.tidyverse.org` (abgerufen am 08.11.2022).

Wuertz, D., Setz, T. and Chalabi, Y. (2022). *fBasics: Rmetrics - Markets and Basic Statistics*. R package version 4021.92. Unter: `https://CRAN.R-project.org/package=fBasics` (abgerufen am 08.11.2022).

Yancey, W. E., Winkler, W. E., and Creecy, R. H. (2002). Disclosure risk assessment in perturbative microdata protection, *in* J. Domingo-Ferrer (ed.), *Inference Control in Statistical Databases: from theory to practice, LNCS 2316*, Springer-Verlag Berlin Heidelberg 2002, p. 135–152.

Yi, G. Y. (2017). *Statistical Analysis with Measurement Error or Misclassification: Strategy, Method and Application*, Springer Series in Statistics.

Elektronischer Anhang

Alle Daten, Codes, Figuren und Ergebnisse, die während der Masterthesis zur Anwendung kamen, sind in elektronischer Form unter folgendem Link bereitgestellt:
`https://drive.google.com/drive/folders/15kx1FsWLYA2wjWFZgtqrGXJNuUGnUgNJ?usp=sharing`

Abbildungsverzeichnis

1 Trade-off zwischen Informationsverlust und Disclosure Risk (aus Templ (2017, S. 43) entnommen) . . . 7
2 Arten von Variablen (aus Templ (2017, S. 36) entnommen) . . . 7
3 Wahl eines Risiko-Thresholds für Local Suppression (aus Templ (2017, S. 108) entnommen) . . . 16
4 Ablauf $MDAV$ Methode für Microaggregation (aus Templ (2017, S. 122) entnommen) . . . 24
5 Ablauf PCA Methode für Microaggregation (aus Templ (2017, S. 121) entnommen) . . . 26
6 Vergleich der Kovarianzen zwischen *pca* und *clustpppca* Methoden (in Blau) mit den (in Schwarz) originalen Daten (aus Templ (2006, S. 355) entnommen) 28
7 Gleiche Kovarianzen (im engeren Sinne) zwischen originalen und perturbierten Daten mit additiver Noise Perturbation (aus Templ (2017, S. 126) entnommen) . . . 31
8 Frequenzverteilung einer metrischen Variable vor und nach additiver Noise in Bezug auf die Varianzstruktur (aus Benschop and Welch (2019) entnommen) . . . 31
9 Menge an Noise und deren Auswirkungen auf den Wertebereich (Perzentile)(aus Benschop and Welch (2019) entnommen) . . . 32
10 Proportionale Kovarianzstruktur zu originalen Daten mit *correlated* Noise Perturbation, mit der Normalverteilungsannahme (aus Templ (2017, S. 127) entnommen) . . . 34
11 Kovarianzstruktur zu originalen Daten mit *correlated2* Noise Perturbation, ohne die Normalverteilungsannahme (aus Templ (2017, S. 128) entnommen) 35
12 Darstellung des Multi-Layer Perceptrons (aus Bonaccorso (2018, S. 328) entnommen) . . . 44
13 Korrelationsplot der EUSILC Daten . . . 48
14 Verteilung der Variable hsize . . . 50
15 Cooks Distanzen der EUSILC Daten . . . 51
16 Perturbation einer binären Zielvariable . . . 55
17 Animierter Vergleich der Variable Importance (> 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei MDAV Microaggregation (ohne Noise) . . . 65

18 Animierter Vergleich der Variable Importance (> 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *additive* Noise (ohne Microaggregation) 66
19 Animierter Vergleich der Variable Importance (> 10) bei GBM mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *correlated2* Noise (ohne Microaggregation) 67
20 Animierter Vergleich der Variable Importance (> 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei MDAV Microaggregation (ohne Noise) 68
21 Animierter Vergleich der Variable Importance (> 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *additive* Noise (ohne Microaggregation) 69
22 Animierter Vergleich der Variable Importance (> 10) bei RF mit (teilweise simultaner) vs. keine Perturbation verschiedener Variablen bei *correlated2* Noise (ohne Microaggregation) 70
23 Whammy-Effekte der Messfehler in den Kovariablen (aus Carroll et al. (2006, S. 2) entnommen); der oberste Plot stellt die wahre Variable dar, der unterste dagegen die perturbierte 72
24 Darstellung des Attenuation Effekts bei einer perturbierten Kovariablen X^* (aus Yi (2017, S. 47) entnommen) 73
25 Darstellung des Extrapolation-Steps für das SIMEX Modell mit einer quadratischen Approximation 76

Tabellenverzeichnis

1 Beispiel für die Verletzung von K-Anonymität 9
2 Beispiel für die Verletzung von L-Diversität (aus Templ (2017, S. 58) entnommen) 10
3 Beispiel für Recodingverfahren mit kategorialer und metrischer Variable . . 14
4 Beispiel für Local Suppression mit kategorialer Variable, indem " * " den generierten fehlenden Wert bezeichnet 16
5 Beispiel für Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird 21
6 Beispiel für Methode *onedims* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird 22
7 Beispiel für Methode *pca* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird 25
8 Beispiel für Methode *simple* in Microaggregation mit metrischen Variablen, indem 2-Anonymität erreicht wird 28
9 Beispiel für Methode *additive* in Noise Addition mit metrischen Variablen . 30
10 Beispiel für Methode *correlated* in Noise Addition mit metrischen Variablen 33
11 Beispiel für Methode *correlated2* in Noise Addition mit metrischen Variablen 35
12 Beispiel für Methode *outdect* in Noise Addition mit metrischen Variablen . 36
13 Performance Metriken für eine metrische Zielvariable 39
14 Confusion Matrix für eine binäre Zielvariable 39
15 EUSILC Datensatz 47
16 Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable var2 über die 100 Seeds; Parameter γ_{var1} steuert die Korrelation während β_{var2} die Variable Importance bezüglich var2 kontrolliert 61
17 Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable invests über die 100 Seeds; Parameter γ_{age} steuert die Korrelation . . 61
18 Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable age über die 100 Seeds; Parameter γ_{age} steuert die Korrelation während β_{age} die Variable Importance bezüglich age kontrolliert 62
19 Variable Importance vs. Korrelation vs. RMSE hinsichtlich perturbierter Variable var1 über die 100 Seeds; Parameter γ_{var1} steuert die Korrelation während β_{var1} die Variable Importance bezüglich var1 reguliert 62

20 Variable Importance vs. Korrelation vs. RMSE hinsichtlich additiver perturbierter Variablen age, invests und var2 über die 100 Seeds; hier sind sie normalverteilt 63
21 Variable Importance vs. Korrelation vs. RMSE hinsichtlich *correlated2* perturbierter Variable age bzw. var2 über die 100 Seeds; hier sind age und var2 normalverteilt 63
22 Variable Importance vs. Korrelation vs. RMSE hinsichtlich *correlated2* perturbierter Variable age über die 100 Seeds; hier stammt age nicht aus einer Normalverteilung, sondern aus einer Poisson Verteilung 64
23 Disclosure Risk (DR) vs. Informationsverlust (IL) bezüglich drei Anonymisierungsverfahren auf den Variablen mit unterschiedlichen Korrelationsstrukturen; simultane Perturbation aller vier Variablen 71
24 Disclosure Risk (DR) vs. Informationsverlust (IL) bezüglich drei Anonymisierungsverfahren auf den Variablen mit unterschiedlichen Korrelationsstrukturen; gruppenweise Perturbation innerhalb der korrelierten Gruppe 72
25 Vergleich der Ergebnisse der Koeffizienten mittels True-, Naiv- und SIMEX Modell mit X normalverteilt vs. nicht 76
26 Vergleich der Ergebnisse des RMSEs mittels True-, Naiv- und RC (Regressionskalibrierung) Modell mit $var2$ normalverteilt vs. nicht und quadratische vs. lineare Regressionsart 78
27 Vergleich der Accuracy ohne vs. mit der entwickelten Korrektur der individuellen Prädiktionen bei binärer perturbierter Variable 86
28 Vergleich der Accuracy ohne vs. mit der entwickelten Korrektur der individuellen Prädiktionen bei binärer perturbierter Zielvariable 87
29 Vergleich der Accuracy ohne vs. mit der entwickelten (teilweise) Korrektur der individuellen Prädiktionen bei binärer perturbierter Zielvariable; Anwendung nur auf $T_{X^*}(0)$ 87

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Munich, 09.11.2022

Name