Universität Leipzig
Fakultät für Mathematik und Informatik
Mathematisches Institut

Klassifikation mittels adaptiver Partitionierung

Diplomarbeit

Leipzig, November 2016

vorgelegt von
Sambale, Alexander
Diplom Mathematik

Betreuender Hochschullehrer:
Prof. Dr. Max von Renesse
Fakultät für Mathematik und Informatik
Mathematisches Institut

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Aufbau der Arbeit 1
1.2 Einführung in maschinelles Lernen 1

2 Grundlagen für binäre Klassifikation **6**

3 Abschätzung des zusätzlichen Risikos für Mengenschätzer **12**
3.1 Obere Schranke für den Schätzfehler 12
3.2 Abschätzung des Modulus durch Tsybakov-Bedingungen 25
3.3 Obere Schranke für den Näherungsfehler 33

4 Modellauswahl **37**

5 Anwendung auf Partitionierungen mittels dyadischen Bäumen **41**

6 Implementierung von CUDTP und ODT **44**
6.1 Classification Using Dyadic Tree Partitioning (CUDTP) 44
6.2 Optimal Dyadic Decision Tree (ODT) 51

7 Vergleich mit typischen Verfahren zur Klassifizierung **58**
7.1 Typische Verfahren 58
7.1.1 Lineare Regression 58
7.1.2 Logistische Regression (Logit) 61
7.1.3 Lineare Diskriminanzanalyse (LDA) 62
7.1.4 Quadratische Diskriminanzanalyse (QDA) 64
7.1.5 Support Vector Machines (SVM) 64
7.1.6 Entscheidungsbäume (Tree) 68
7.1.7 Random Forest (RF) 70
7.1.8 k-Nächste-Nachbarn-Klassifikation (KNN) 70
7.1.9 Neuronale Netze (NNet) 71
7.1.10 Kreuzvalidierung 73
7.1.11 Dimensionsreduzierung 74

7.1.12 Hauptkomponentenanalyse 74
7.2 Vergleich verschiedener Verfahren mit Hilfe von R 75
7.2.1 Kreis 76
7.2.2 Schachbrettmuster 78
7.2.3 Diagonale Fläche 79
7.2.4 Datensatz aus der Praxis 82
7.2.5 Zusammenfassung des Vergleichs 87

Kurzzusammenfassung **88**

Literaturverzeichnis **92**

Abbildungsverzeichnis **93**

Tabellenverzeichnis **94**

Programmauszüge **95**

Liste der Algorithmen **96**

A Abschätzungen und Näherungen **97**

B Kenngrößen **113**

C Verwendete Hard- und Software **114**
C.1 R 114
C.2 Anderes 114

Erklärung **115**

1 Einleitung

1.1 Aufbau der Arbeit

Diese Arbeit beruht auf der Veröffentlichung „Classification algorithms using adaptive partitioning“ von Binev et al. [Bin+14a]. Dabei sollen die Ideen und Beweise nachvollzogen werden. Ziel ist es dabei auch, am Ende eine Implementierung des in Kapitel 6 von [Bin+14a, S. 2156] vorgestellten Algorithmus zur Erstellung von dyadischen Entscheidungsbäumen zu erhalten. Im gleichen Atemzug wird ein weiterer Algorithmus basierend auf dyadischen Entscheidungsbäumen implementiert, vgl. [Bla+07, S. 8]. Zum Schluss werden die daraus resultierenden Klassifikatoren mit typischen Klassifikationsalgorithmen unter Zuhilfenahme von R für ausgewählte Problemstellungen verglichen.

1.2 Einführung in maschinelles Lernen

Nach [Lan15, S. 3] ist maschinelles Lernen der Forschungsbereich, der sich mit der Entwicklung von Computeralgorithmen beschäftigt, die Daten in intelligente Anweisungen oder begründete Schlussfolgerungen umwandeln. Dieser Bereich wird dabei stark von den verfügbaren Daten, der Rechenleistung und den verwendeten statistischen Verfahren beeinflusst.

Maschinelles Lernen spielt in vielen Arealen der Wissenschaft, Finanzwelt und Industrie eine Hauptrolle, siehe dazu [HTF09, S. 1 ff.]. Zugehörige Beispiele sind:

- Das Vorhersagen, ob ein Patient, der bereits einen Herzinfarkt hatte, einen zweiten erleiden wird. Dabei basiert die Prognose auf der Ernährungsweise, sowie bevölkerungsstatistischen und klinischen Messungen.
- Die Vorhersage, welchen Preis eine Aktie in 6 Monaten haben wird, mit Kenntnis der wirtschaftlichen Daten einer Firma.
- Die Erkennung einer handgeschriebenen Postleitzahl von einem digitalisierten Bild.
- Eine Schätzung des Glukosewertes im Blut eines Diabetes Patienten anhand der durch das Blut absorbierten infraroten Strahlung.
- Die Beurteilung, ob eine Email als Spam eingeordnet werden soll oder nicht.

Ein Begriff, der häufig mit maschinellem Lernen in Verbindung gebracht wird, ist Data-Mining. Dabei wird versucht aus großen Datenbanken neue Erkenntnisse zu extrahieren. In großen Teilen überschneiden sich die beiden Bereiche. Jedoch liegt die Aufmerksamkeit bei maschinellem Lernen vor allem darauf, dem Computer beizubringen, wie man Daten benutzt, um eine Aufgabe zu lösen. Hingegen versucht man bei Data-Mining, dem Computer Mustererkennung zu lehren. Diese wird wiederum von Menschen benutzt, um verschiedene Problemlösungen zu erhalten, vgl. [Lan15, S. 3].

Laut [Lan15, S. 9 ff.] kann man den Prozess des Lernens in vier zusammenhängende Komponenten einteilen:

- Bei der *Datenspeicherung* werden Beobachtungen, Speicher und Erinnerungen benutzt, um eine sachliche Basis für weitergehende Schlussfolgerungen zu erhalten.
- *Abstraktion* beinhaltet die Übersetzung der gespeicherten Daten in umfassendere Darstellungen und Konzepte.
- Der Vorgang der *Verallgemeinerung* verwendet abstrahierte Daten zur Schaffung von Wissen und beeinflusst die Handlungsanweisungen in neuen Zusammenhängen.
- Die *Auswertung* liefert einen Rückmeldungsmechanismus, um die Nützlichkeit des gelernten Wissens zu messen und informiert über potentielle Verbesserungen.

Zur Datenspeicherung gehört auch, die Daten entsprechend der Verwendung aufzubereiten. Dazu muss man wissen, wie diese strukturiert sein sollten. Bei der Untersuchung eines Sachverhaltes werden Eigenschaften von Interesse gemessen. Diese Eigenschaften nennt man *Merkmale* und eine spezielle Ausprägung aller gemessenen Merkmale heißt *Beobachtung*. Typischerweise werden die Beobachtungen und deren Merkmale in Matrixform gespeichert, wobei die Beobachtungen die Zeilen und die Merkmale die Spalten bilden. Ein Merkmal wird *numerisch* genannt, wenn es in Zahlen gemessen wird. Kann es nur einen Wert aus einer Menge von Kategorien annehmen, so nennt man es *kategorial* oder auch *nominal*. Liegen die Kategorien in einer geordnete Liste vor, so nennt man das Merkmal *ordinal*. Die Art und Anzahl der Merkmale des Datensatzes schränkt dabei die Auswahl der geeigneten Maschinen-Lern-Algorithmen ein.

Im Schritt der Abstraktion werden die Beobachtungen und deren Merkmalsausprägungen verwendet, um Wissen zu erwerben. Dazu wird intern eine explizite

Darstellung der in den Daten enthaltenen Muster erstellt. Diese Repräsentation nennt man auch *Modell*. Beispiele für Modelle sind mathematische Gleichungen, relationale Diagramme, Bäume, Graphen, sowie Wenn-Dann-Regeln, aber auch Gruppierungen von Daten. Die Auswahl des Modells wird dabei je nach Anwendungsfall vom Benutzer getroffen, geschieht also nicht maschinengesteuert. Steht fest, welches Modell benutzt werden soll, so muss es noch an den Datensatz angepasst werden. Diesen Vorgang nennt man *Training*. Hat man ein Modell trainiert, so liegen die Daten umgewandelt in abstrahierter Form vor und geben die ursprünglichen Informationen zusammenfassend wieder. Ein gelerntes Modell liefert allerdings keine neuen Daten, sondern gibt erworbenes Wissen wieder.

Das Verallgemeinern beschreibt in diesem Zusammenhang den Prozess der Umwandlung des abstrakten Wissens in eine nutzbare Form. Man versucht, die erkannten Muster auf eine kleine, relevante Anzahl zu reduzieren. Im Endeffekt ergibt sich daraus ein Algorithmus. Dieser wendet eine Heuristik an, um fundierte Vermutungen anzustellen, wie man die besten Schlussfolgerungen findet.

Da dies nur Vermutungen sind, entstehen wie auch beim Menschen fehlerhafte Schlussfolgerungen. Sind die Rückschlüsse methodisch fehlerbehaftet oder falsch in einer vorhersagbaren Art und Weise, so nennt man den Algorithmus verzerrt. Er hat eine *Verzerrung*.

Verzerrung ist ein notwendiges Übel, welches zum Prozess der Abstraktion und Verallgemeinerung dazugehört und somit jeder Lernaufgabe innewohnt. Jeder Lerner ist auf besondere Art verzerrt und hat somit seine Schwächen. Deswegen muss man noch auswerten oder messen wie groß der Erfolg des Lernenden ungeachtet seiner Verzerrungen ist. Stellt man fest, dass der Vorgang des Lernens nicht erfolgreich genug war, so kann erneutes Training erforderlich sein. Die Auswertung erfolgt normalerweise, nachdem ein Modell auf einem Anfangsdatensatz trainiert wurde. Danach wird das Modell auf einen Testdatensatz angewendet, um zu entscheiden, wie gut die Verallgemeinerung auf neuen, ungesehen Daten ausfällt.

Nur in extrem seltenen Fällen wird das Modell eine perfekte Verallgemeinerung der Daten liefern. Das liegt an einem Phänomen, das als *Rauschen* bezeichnet wird. Es entsteht durch unerklärbare Schwankungen in den Daten. Verrauschte Daten ergeben sich durch scheinbar zufällige Ereignisse bei der Erhebung und Verarbeitung der Daten:

- Messfehler von Sensoren
- Umwandlung von Daten, bei denen Informationen verloren gehen
- Umfragen, die nur Antworten in engem Rahmen zulassen
- unberücksichtigte komplexe Vorgänge, die unsystematisch erscheinen

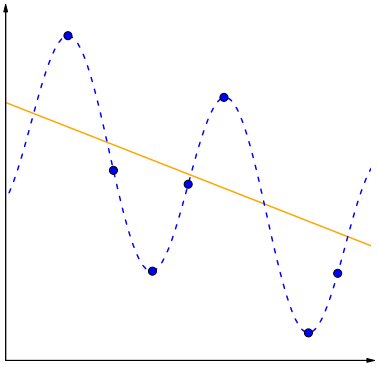


Abbildung 1: Beispiel zu Überanpassung. Die blauen Punkte stellen Datenpunkte dar, die durch Rauschen aus dem tatsächlichen Muster in Orange entstanden sind. Die gestrichelte blaue Linie ist dabei ein komplexes Muster, welches aus den Datenpunkten rekonstruiert wurde und zu Überanpassung führt.

Der Versuch das Rauschen der Daten abzubilden ist ein Problem, welches *Überanpassung* genannt wird. Da das Rauschen per Definition unerklärbar ist, wird der Versuch es zu interpretieren in fehlerbehafteten Schlussfolgerungen enden, die sich schlecht auf neue Fälle verallgemeinern lassen. Solche Anstrengungen haben typischer Weise komplexere Modelle zur Konsequenz und diese werden das wahre Muster übersehen, welches der Lerner versucht zu erkennen. Ein Modell, das während des Trainings gut abzuschneiden scheint, aber bei der Auswertung schlechte Ergebnisse liefert, nennt man an den Trainingsdatensatz überangepasst, denn es verallgemeinert unzureichend auf dem Testdatensatz, siehe dazu auch Abbildung 1.

Das Problem der Verzerrung und des Rauschen sollte man im Hinterkopf behalten, weil das Lösen dieser Schwierigkeit eine wichtige Quelle zur Unterscheidung und somit zur Wahl der geeigneten Modelle darstellt.

In diesem Zusammenhang stellt sich die Frage, warum man überhaupt verschiedene Verfahren benutzt und ob es nicht eine beste Methode gibt, welche für jedes Problem angewendet werden sollte. Die Antwort auf diese Überlegung ist, dass es zu jedem Modell immer andere Modelle gibt die auf bestimmten Problemen besser abschneiden. Für diesen Sachverhalt wird in [Lan15, S. 15] auf das „No Free Lunch Theorem“ verwiesen und auch in der Einleitung von [DGL96, S. 5] erfolgt ein Hinweis auf diese Problematik. Es gibt also kein universell bestes Verfahren, sondern es ist immer von der jeweiligen Aufgabe abhängig. Deswegen sind Vergleiche von Modellen anhand von Beispielen nur bedingt verallgemeinerungsfähig und man versucht Methoden zu finden, die für viele in der Praxis vorkommende Problemklassen möglichst gute Ergebnisse liefern.

In [Jam+16, S. 373] werden zwei Gruppen von Lernmethoden, das *überwachte* und *unüberwachte Lernen*, untersucht.

Beim überwachten Lernen hat man Zugriff auf eine Menge von d Merkmalen $X_1, \ldots, X_d$, auch Prädiktoren genannt und ein Zielmerkmal Y, die auf n Beobachtungen gemessen werden. Ziel ist es dabei Y in Abhängigkeit der $X_1, \ldots, X_d$ vorherzusagen. Der Wert des Zielmerkmals verschafft dabei eine Möglichkeit zu überprüfen, wie erfolgreich das Lernen war. Daher kommt auch der Name überwachtes Lernen. Nimmt das Zielmerkmal numerische Werte an, so wird die Vorhersageaufgabe *Regression* genannt. Bei nominalem Zielmerkmal bezeichnet man diese Aufgabe als *Klassifikation*. Für Beispiele solcher Methoden sei auf Abschnitt 7.1 verwiesen.

Beim unüberwachten Lernen gibt es kein Zielmerkmal. Die Aufgabe dieses Lerntyps besteht darin, interessante Sachverhalte über die Messungen $X_1, \ldots, X_d$ herauszufinden. Dazu versucht man, die Daten in aufschlussreicher Art darzustellen. Ein Beispiel dafür ist die Suche nach einer Unterteilung in Gruppen, das sogenannte Clustering.

Im weiteren Verlauf der Arbeit wird für uns jedoch nur das überwachte Lernen, genauer die binäre Klassifikation, eine Rolle spielen.

2 Grundlagen für binäre Klassifikation

Die Diplomarbeit beschränkt sich auf das Thema der binären Klassifikation, ein Teilgebiet des überwachten Lernens. Dabei werden sich die Bezeichnungen und Ideen stark an der Vorgabe von [Bin+14a] orientieren.

Im Fall der *binären Klassifikation* fasst man die d Merkmale $X_1, \dots, X_d$ aus der Einleitung zu einer Variablen $X \in \mathbf{X} \subseteq \mathbb{R}^d$ zusammen und setzt $\mathbf{Y} = \{-1, 1\}$ mit dem Zielmerkmal $Y \in \mathbf{Y}$. Weiter sei $\mathbf{Z}$ definiert als $\mathbf{Z} = \mathbf{X} \times \mathbf{Y}$ und $\rho = \rho_{\mathbf{X}}(X) \cdot \rho(Y|X)$ sei ein Wahrscheinlichkeitsmaß auf $\mathbf{Z}$. Dazu seien *Beobachtungsdaten* $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = (z_i)_{i=1}^n, z_i = (x_i, y_i)$ mit $i = 1, \dots, n$ unabhängig bzgl. ρ verteilt. Darüber hinaus bezeichne $p(x)$ die Wahrscheinlichkeit für $Y = 1$ gegeben $X = x$, $\mathbf{E}$ den Erwartungswert und $\mathbf{Var}$ die Varianz. Die Definition der Grundbegriffe kann man unter anderem in [Kle13] nachlesen.

Definition 2.1. Die *Regressionsfunktion* wird definiert durch

$$\eta(x) := \mathbf{E}(Y|X = x).$$

Da Y nur die Werte -1 und 1 annehmen kann, ergibt sich nach Anwendung der entsprechenden Definitionen:

$$\begin{aligned} \eta(x) &= \mathbf{E}(Y|X = x) \\ &= 1 \cdot \mathbf{P}(Y = 1|X = x) - 1 \cdot \mathbf{P}(Y = -1|X = x) \\ &= p(x) - (1 - p(x)) \\ &= 2p(x) - 1. \end{aligned}$$

Weitere Begriffe werden neben [Bin+14a] auch aus [DGL96] entnommen.

Definition 2.2. Eine Abbildung $g\colon \mathbf{X} \to \mathbf{Y}$ mit $\mathbf{X} \subseteq \mathbb{R}^d$ und $\mathbf{Y} = \{-1, 1\}$ heißt *binärer Klassifikator* oder auch *binärer Klassifizierer*.

Ein solcher Klassifizierer g gibt den Wert $y = 1$ zurück, falls x in einer gewissen Menge $\Omega \subseteq \mathbf{X}$ liegt und ansonsten $y = -1$. Deswegen lässt er sich als

$$g = \mathbb{1}_\Omega - \mathbb{1}_{\Omega^c}$$

schreiben und ist durch diese Darstellung eindeutig bestimmt. Dabei bezeichne $\mathbb{1}_\Omega$ die Indikatorfunktion, die für $\omega \in \Omega$ den Wert 1 und für $\omega \notin \Omega$ den Wert 0 annimmt. Ferner ist $\Omega^c = \mathbf{X} \backslash \Omega$ das Komplement von Ω bzgl. $\mathbf{X}$. Um die Abhängigkeit des

Klassifikators von der Menge Ω zu verdeutlichen, schreibt man statt g auch g_Ω und später werden wir stellvertretend für den Klassifikator nur die Menge Ω verwenden. Sei von hieran eine solche Menge Ω immer $\rho_{\mathbf{X}}$ messbar.

Ein Klassifizierer g begeht einen Fehler, falls $g(x) \neq y$ gilt und seine Fehlerwahrscheinlichkeit ist gegeben durch

$$\mathbf{P}[g(X) \neq Y].$$

Definition 2.3. Der *Bayes'sche Klassifizierer* bzgl. der Verteilung ρ wird definiert durch

$$g^*(x) := g_{\Omega^*}(x) := \begin{cases} 1 & \text{wenn } \eta(x) \geq 0, \\ -1 & \text{sonst.} \end{cases}$$

Hierbei ist $\Omega^* := \{x \in \mathbf{X} \colon \eta(x) \geq 0\}$.

Definition 2.4. Als *Risiko* des Klassifizierers g_Ω bezeichnen wir die Wahrscheinlichkeit der falschen Klassifikation

$$R(\Omega) := R(g_\Omega) := \int_{\mathbf{X}} \mathbf{P}[g_\Omega(X) \neq Y] \, \mathrm{d}\rho_{\mathbf{X}}.$$

Für eine messbare Menge $S \subseteq \mathbf{X}$ führen wir als weitere Notation

$$\rho_S := \rho_{\mathbf{X}}(S) = \int_S \mathrm{d}\rho_{\mathbf{X}} \qquad \text{und} \qquad \eta_S := \int_S \eta \, \mathrm{d}\rho_{\mathbf{X}} \tag{2.1}$$

ein. Aus

$$x \in S \Rightarrow g_S(x) = 1 \qquad \text{und} \qquad x \notin S \Rightarrow g_S(x) = -1$$

folgt dann

$$\begin{aligned} R(S) &= \int_S \mathbf{P}[g_S(X) \neq Y] \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S^c} \mathbf{P}[g_S(X) \neq Y] \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_S (1-p) \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S^c} p \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_{\mathbf{X}} p \, \mathrm{d}\rho_{\mathbf{X}} - \int_S p \, \mathrm{d}\rho_{\mathbf{X}} + \int_S (1-p) \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_{\mathbf{X}} p \, \mathrm{d}\rho_{\mathbf{X}} - \int_S \eta \, \mathrm{d}\rho_{\mathbf{X}}. \end{aligned} \tag{2.2}$$

Aus dieser Gleichung ergibt sich, dass der Bayes'sche Klassifizierer minimales Risiko $R(\Omega^*)$ besitzt und alle anderen Klassifikatoren mit minimalem Risiko unterscheiden sich nur auf Mengen vom Maße Null oder auf denen η verschwindet.

In Anwendungen wollen wir minimales Risiko. Der Bayes'sche Klassifikator ist in dieser Hinsicht optimal. Jedoch ist die Verteilung und somit der Bayes'sche Klassifikator unbekannt.

Definition 2.5. Der Term

$$R(\Omega) - R(\Omega^*)$$

wird als *zusätzliches Risiko* bezeichnet.

Für einen zu konstruierenden Klassifizierer g_Ω, versucht man diesen Term so klein wie möglich werden zu lassen. Um das empirisch zu erreichen, wird eine Familie $\mathcal{S}$ von Mengen vorgegeben, aus der Ω ausgewählt wird. Klassifikationsmethoden, die auf solch einer Strategie aufbauen, nennt man auch *Mengenschätzer*. Wir benutzen die Schreibweise

$$\Omega_\mathcal{S} := \operatorname*{argmin}_{S\in\mathcal{S}} R(S)$$

für die Menge aus der Familie $\mathcal{S}$, die das zusätzliche Risiko minimiert. Auch diese Menge $\Omega_\mathcal{S}$ ist uns unbekannt, jedoch dient sie als Ziel, um herauszufinden, wie gut ein Klassifizierer unter Benutzung dieser Familie $\mathcal{S}$ bestenfalls sein kann. Das erste Integral nach dem letzten Gleichheitszeichen in (2.2) ist unabhängig von S. Also wird $R(S)$ minimiert, wenn das zweite Integral nach dem letzten Gleichheitszeichen von (2.2) maximiert wird. Damit erhält man

$$\Omega_\mathcal{S} = \operatorname*{argmin}_{S\in\mathcal{S}} R(S) = \operatorname*{argmax}_{S\in\mathcal{S}} \eta_S. \tag{2.3}$$

Mit der Definition von $\Omega_\mathcal{S}$ lässt sich das zusätzliche Risiko von S in zwei nichtnegative Teile auftrennen:

$$R(S) - R(\Omega^*) = \Big(R(S) - R(\Omega_\mathcal{S})\Big) + \Big(R(\Omega_\mathcal{S}) - R(\Omega^*)\Big).$$

Definition 2.6. Der Ausdruck

$$R(S) - R(\Omega_\mathcal{S})$$

wird als *Schätzfehler* bezeichnet.

Definition 2.7. Der *Näherungsfehler* wird definiert durch:

$$a(\Omega^*, \mathcal{S}) := R(\Omega_\mathcal{S}) - R(\Omega^*).$$

Ein Klassifizierungsalgorithmus benutzt die Ziehung der Beobachtungsdaten $\mathbf{z}$, um empirisch eine Menge $\hat{\Omega} \in \mathcal{S}$ zu finden, aus der ein Klassifikator konstruiert wird. Damit ist der Schätzfehler $R(\hat{\Omega}) - R(\Omega_{\mathcal{S}})$ eine Zufallsvariable, die von den Beobachtungsdaten, dem numerischen Verfahren zur Berechnung von $\hat{\Omega}$ und der Komplexität bzw. der Größe von $\mathcal{S}$ abhängt. Im Gegensatz dazu erweist sich der Näherungsfehler nicht als Zufallsvariable, denn der Bayes'sche Klassifizierer hängt nicht von den Beobachtungsdaten ab und die Familie $\mathcal{S}$ ist unabhängig von der konkreten Ziehung der Beobachtungsdaten. Allerdings lässt man eine Abhängigkeit der Familie $\mathcal{S}$ von der Anzahl n der Beobachtungen zu. Man bildet üblicherweise eine verschachtelte Folge $(\mathcal{S}_m)_{m\geq 1}$ von Familien von Mengen mit $\mathcal{S}_m \subseteq \mathcal{S}_{m+1}$ für jedes m. Dabei ist m eine Funktion in Abhängigkeit von n. Die Wahl von m wird hierbei mit dem Ziel die beiden Fehler auszubalancieren getroffen. Mit wachsendem m verringert sich der Näherungsfehler und normalerweise erhöht sich der Schätzfehler.

Mit (2.3) ergibt sich für einen Schätzer $\hat{\eta}_S$ eine natürliche Art einen Klassifizierer aus $\mathcal{S}$ auszuwählen:

$$\hat{\Omega} := \hat{\Omega}_{\mathcal{S}} := \operatorname*{argmax}_{S\in\mathcal{S}} \hat{\eta}_S. \tag{2.4}$$

Man beachte, dass das Maximum nicht notwendigerweise eindeutig sein muss. Eine der gebräuchlichsten Vorgehensweisen, um $\hat{\eta}_S$ zu bauen, ist die Einführung der empirischen Gegenstücke zu (2.1),

$$\bar{\rho}_S := \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_S(x_i) \qquad \text{und} \qquad \bar{\eta}_S := \frac{1}{n}\sum_{i=1}^{n} y_i \mathbb{1}_S(x_i). \tag{2.5}$$

Das empirische Risiko wird wie folgt definiert:

$$\bar{R}(S) := \frac{1}{n}\left|\left\{i \in \{1,\dots,n\} \colon g_S(x_i) \neq y_i\right\}\right|. \tag{2.6}$$

Dabei bezeichnet $|M|$ für eine Menge M die Anzahl der Elemente dieser Menge. Wählt man $\hat{\eta}_S = \bar{\eta}_S$ bei der Suche nach $\hat{\Omega}$, so ist das Maximieren von $\hat{\eta}_S$ äquivalent zum Minimieren des empirischen Risikos $\bar{R}(S)$ über die Familie $\mathcal{S}$. Deswegen ergibt sich in diesem Fall für $\hat{\Omega} = \bar{\Omega}$:

$$\bar{\Omega}_{\mathcal{S}} = \operatorname*{argmin}_{S\in\mathcal{S}} \bar{R}(S).$$

Andere Wahlen von $\hat{\eta}$ liefern entsprechend andere Klassifikatoren.

Im Folgenden werden immer Ergebnisse gesucht, die mit hoher Wahrscheinlichkeit gelten, anstatt in Erwartung. Denn aus Abschätzungen der Wahrscheinlichkeit

in der Form

$$\mathbf{P}\left[R(\hat{\Omega}) - R(\Omega^*) \geq C_0 n^{-\alpha}\right] \leq C_1 n^{-r}$$

lassen sich auch nachfolgende Schranken für den Erwartungswert ableiten:

$$\mathbf{E}\left[R(\hat{\Omega}) - R(\Omega^*)\right] \leq C_0 n^{-\alpha} + C_1 n^{-r}.$$

Das ergibt sich aus dem folgenden Satz mit $X = R(\hat{\Omega}) - R(\Omega^*)$, da $R(\hat{\Omega}) - R(\Omega^*) \leq 1$ gilt.

Satz 2.8. *Gelte* $\mathbf{P}[X \geq \varepsilon] \leq \delta$ *und* $X \leq 1$ *für* $\varepsilon, \delta > 0$*, dann ist* $\mathbf{E}[X] \leq \delta + \varepsilon$*.*

Beweis. Es gilt:

$$\begin{aligned}
\mathbf{E}[X] &= \int X \, d\mathbf{P} \\
&= \int \mathbb{1}_{\{X \geq \varepsilon\}} X + \mathbb{1}_{\{X < \varepsilon\}} X \, d\mathbf{P} \\
&\leq \int \mathbb{1}_{\{X \geq \varepsilon\}} \, d\mathbf{P} + \int \mathbb{1}_{\{X < \varepsilon\}} \varepsilon \, d\mathbf{P} \\
&= \mathbf{P}[X \geq \varepsilon] + \mathbf{P}[X < \varepsilon]\varepsilon \\
&\leq \delta + \varepsilon.
\end{aligned}$$

□

Die vier hier vorgenommenen Definitionen stammen aus [DGL96, S. 196] bzw. [Gyö+02, S. 143 f.]. Dabei wird der Begriff der Vapnik-Chervonenkis Dimension im Verlaufe der Arbeit eine entscheidende Rolle spielen.

Definition 2.9.

- Sei $\mathcal{A} \subseteq \mathfrak{P}(\mathcal{R}^d)$. Für $z_1, \ldots, z_l \in \mathcal{R}^d$ definiere

$$N_{\mathcal{A}}(z_1, \ldots, z_l) := \left|\{A \cap \{z_1, \ldots, z_l\} \colon A \in \mathcal{A}\}\right|.$$

- Der l-te *Splitterkoeffizient* von $\mathcal{A}$ ist

$$s(\mathcal{A}, l) := \max_{z_1, \ldots, z_l \in \mathcal{R}^d} N_{\mathcal{A}}(z_1, \ldots, z_l).$$

- Ist $\mathcal{A} \neq \emptyset$, so wird mit $V_{\mathcal{A}}$ die *Vapnik-Chervonenkis Dimension* von $\mathcal{A}$ bezeichnet und definiert durch

$$V_{\mathcal{A}} := VC(\mathcal{A}) := \sup\{l \in \mathbb{N} \colon s(\mathcal{A}, l) = 2^l\}. \tag{2.7}$$

Nun folgt eine einfache Folgerung aus der Definition, vgl. [DGL96, S. 219].

Lemma 2.10. *Für eine Mengensystem $\mathcal{A}$ bestehend aus endlich vielen Mengen gilt*

$$s(\mathcal{A}, l) \leq |\mathcal{A}|$$

und

$$V_{\mathcal{A}} \leq \log_2 |\mathcal{A}|.$$

Beweis. Trivialerweise gilt per Definition von $N_{\mathcal{A}}$ für $z_1, \ldots, z_l \in \mathcal{R}^d$ dann

$$N_{\mathcal{A}}(z_1, \ldots, z_l) \leq |A|$$

und somit per Definition von $s(\mathcal{A}, l)$ die Ungleichung

$$s(\mathcal{A}, l) \leq |\mathcal{A}|.$$

Ferner erhält man

$$2^{V_{\mathcal{A}}} = s(\mathcal{A}, V_{\mathcal{A}}) \leq |\mathcal{A}|$$

und da der Logarithmus monoton wachsend ist, sowie $\ln 2 > 0$ gilt, folgt daraus:

$$V_{\mathcal{A}} \leq \log_2 |\mathcal{A}|.$$ □

3 Abschätzung des zusätzlichen Risikos für Mengenschätzer

Die in diesem Abschnitten folgenden Sätze, Lemmata, Definitionen, Folgerungen und Ideen stammen größtenteils aus [Bin+14a, S. 2145 ff.]. Die zugehörigen Beweise wurden um weggelassene Argumente ergänzt bzw. ausformuliert.

3.1 Obere Schranke für den Schätzfehler

Es wird sich in Satz 3.4 zeigen, dass der nachfolgende Satz ein wichtiges Resultat zur Abschätzung des Schätzfehlers liefert. Wegen seiner Wichtigkeit, wird der Beweis aus [Bin+14b], hier detaillierter geführt.

Im Folgenden bezeichne $A \Delta B := (A \backslash B) \cup (B \backslash A)$ die *symmetrische Differenz* für Mengen A und B .

Satz 3.1. *Für hinreichend großes $B > 0$ gilt das Folgende. Ist $\mathcal{S}$ eine Familie von $\rho_{\mathbf{X}}$-messbaren Mengen $S \subseteq \mathbf{X}$ mit endlicher VC Dimension $V_{\mathcal{S}}$ und*

$$e_n(S) := \sqrt{\rho_{S \Delta \Omega_{\mathcal{S}}} \cdot \varepsilon_n} + \varepsilon_n, \qquad \varepsilon_n := \varepsilon_{n,r} := B \cdot \max\{r+1, V_{\mathcal{S}}\} \cdot \frac{\ln n}{n}, \tag{3.1}$$

mit $r > 0$ frei wählbar. Dann existiert eine absolute Konstante C_0, sodass für jedes $n \geq 2$ mit Wahrscheinlichkeit mindestens $1 - C_0 n^{-r}$ auf den Daten $\mathbf{x} \in \mathbf{X}^n$ für jedes $S \in \mathcal{S}$ die folgende Abschätzung gilt:

$$|\eta_S - \eta_{\Omega_{\mathcal{S}}} - (\bar{\eta}_S - \bar{\eta}_{\Omega_{\mathcal{S}}})| \leq e_n(S). \tag{3.2}$$

Beweis. Um die Aussage des Satzes zu beweisen, werden wir für eine Familie $\mathcal{S}$ von Mengen $S \subseteq \mathbf{X}$ mit endlicher VC Dimension $V_{\mathcal{S}}$ die folgende Wahrscheinlichkeit abschätzen:

$$\mathbf{P}\left[\exists S \in \mathcal{S} \colon\ |\eta_{\Omega_{\mathcal{S}}} - \eta_S - (\bar{\eta}_{\Omega_{\mathcal{S}}} - \bar{\eta}_S)| > e_n(S)\right].$$

Dabei ist

$$e_n(S) := \sqrt{\rho_{S \Delta \Omega_{\mathcal{S}}} \cdot \varepsilon_n} + \varepsilon_n, \qquad \varepsilon_n := K \cdot \frac{\ln n}{n}, \qquad K := B \cdot \max\{r+1, V_{\mathcal{S}}\}.$$

Mit einer hinreichend großen Konstanten $B > 0$ soll diese Wahrscheinlichkeit klein werden.

Für $S \in \mathcal{S}$ definieren wir dazu $f_S(x,y) := y(\mathbb{1}_{\Omega_\mathcal{S}}(x) - \mathbb{1}_S(x))$ mit $x \in \mathbf{X}$ und $y \in \mathbf{Y}$ und zeigen für uns später nützliche Eigenschaften. Sei (X,Y) bzgl. ρ verteilt, so gilt für den Erwartungswert der Zufallsvariablen $f_S(X,Y)$:

$$\begin{aligned}\mathbf{E}\left[f_S(X,Y)\right] &= \mathbf{E}\left[Y\left(\mathbb{1}_{\Omega_\mathcal{S}}(X) - \mathbb{1}_S(X)\right)\right] = \int_{\mathbf{Z}} Y\left(\mathbb{1}_{\Omega_\mathcal{S}}(X) - \mathbb{1}_S(X)\right) \mathrm{d}\rho \\ &= \int_{\mathbf{X}} \left(\mathbf{P}[Y=1|X=x] - \mathbf{P}[Y=-1|X=x]\right)\left(\mathbb{1}_{\Omega_\mathcal{S}}(x) - \mathbb{1}_S(x)\right) \mathrm{d}\rho_{\mathbf{X}}(x) \\ &= \int_{\mathbf{X}} \eta\left(\mathbb{1}_{\Omega_\mathcal{S}} - \mathbb{1}_S\right) \mathrm{d}\rho_{\mathbf{X}} = \int_{\Omega_\mathcal{S}} \eta \,\mathrm{d}\rho_{\mathbf{X}} - \int_S \eta \,\mathrm{d}\rho_{\mathbf{X}} \\ &= \eta_{\Omega_\mathcal{S}} - \eta_S. \end{aligned} \tag{3.3}$$

Außerdem ist per Definition in (2.5)

$$\frac{1}{n}\sum_{j=1}^{n} f_S(x_j, y_j) = \bar{\eta}_{\Omega_\mathcal{S}} - \bar{\eta}_S.$$

Für alle $x \in \mathbf{X}$ und $y \in \mathbf{Y}$ gilt wegen $\mathbf{Y} = \{-1, 1\}$ und $\mathbb{1}_{\Omega_\mathcal{S}}(x) - \mathbb{1}_S(x) \in \{-1, 0, 1\}$ dann

$$|f_S(x,y)| = |y\left(\mathbb{1}_{\Omega_\mathcal{S}}(x) - \mathbb{1}_S(x)\right)| \leq |y| \cdot |\mathbb{1}_{\Omega_\mathcal{S}}(x) - \mathbb{1}_S(x)| \leq 1 \cdot 1 = 1. \tag{3.4}$$

Weiter ist wegen (3.3)

$$\begin{aligned}|\mathbf{E}\left[f_S(X,Y)\right]| &= |\eta_{\Omega_\mathcal{S}} - \eta_S| \\ &= \left|\int_{\Omega_\mathcal{S}} \eta \,\mathrm{d}\rho_{\mathbf{X}} - \int_S \eta \,\mathrm{d}\rho_{\mathbf{X}}\right| = \left|\int_{\mathbf{X}} (\mathbb{1}_{\Omega_\mathcal{S}} - \mathbb{1}_S)\eta \,\mathrm{d}\rho_{\mathbf{X}}\right| \\ &\leq \int_{\mathbf{X}} |(\mathbb{1}_{\Omega_\mathcal{S}} - \mathbb{1}_S)| \cdot |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \leq \int_{\mathbf{X}} |(\mathbb{1}_{\Omega_\mathcal{S}} - \mathbb{1}_S)| \,\mathrm{d}\rho_{\mathbf{X}} \\ &= \rho_{\mathbf{X}}(\Omega_\mathcal{S} \Delta S) \leq 1. \end{aligned} \tag{3.5}$$

Mit der Dreiecksungleichung und den gerade gewonnen Abschätzungen (3.4), (3.5) folgt

$$\begin{aligned}\|f_S(X,Y) - \mathbf{E}[f_S(X,Y)]\|_\infty &\leq \|f_S(X,Y)\|_\infty + \|\mathbf{E}[f_S(X,Y)]\|_\infty \\ &\leq 1 + 1 = 2. \end{aligned} \tag{3.6}$$

Dabei sei für die Definition von $\|\cdot\|_\infty$ auf Definition A.1 verwiesen. Mit der Notation

$$\sigma_S := \sqrt{\mathbf{Var}[f_S(X,Y)]}$$

ergibt sich

$$\begin{aligned}
\sigma_S^2 &= \mathbf{Var}[f_S(X,Y)] = \mathbf{E}[f_S^2(X,Y)] - (\mathbf{E}[f_S(X,Y)])^2 \\
&\leq \mathbf{E}[f_S^2(X,Y)] \\
&= \mathbf{E}\left[(Y\,(\mathbb{1}_{\Omega_\mathcal{S}}(X) - \mathbb{1}_S(X)))^2\right] \\
&= \mathbf{E}\left[Y^2\left(\mathbb{1}_{\Omega_\mathcal{S}}^2(X) - 2\cdot\mathbb{1}_{\Omega_\mathcal{S}}(X)\mathbb{1}_S(X) + \mathbb{1}_S^2(X)\right)\right] \\
&= \mathbf{E}\left[\mathbb{1}_{\Omega_\mathcal{S}}(X) - 2\cdot\mathbb{1}_{\Omega_\mathcal{S}}\mathbb{1}_S(X) + \mathbb{1}_S(X)\right] \\
&= \int_{\mathbf{X}} \mathbb{1}_{\Omega_\mathcal{S}} - 2\cdot\mathbb{1}_{\Omega_\mathcal{S}}\mathbb{1}_S + \mathbb{1}_S \,\mathrm{d}\rho_{\mathbf{X}} \\
&= \int_{\Omega_\mathcal{S}} \mathrm{d}\rho_{\mathbf{X}} - \int_{\Omega_\mathcal{S}\cap S} 2\,\mathrm{d}\rho_{\mathbf{X}} + \int_S \mathrm{d}\rho_{\mathbf{X}} \\
&= \int_{\Omega_\mathcal{S}\setminus(\Omega_\mathcal{S}\cap S)} \mathrm{d}\rho_{\mathbf{X}} + \int_{\Omega_\mathcal{S}\cap S} \mathrm{d}\rho_{\mathbf{X}} - \int_{\Omega_\mathcal{S}\cap S} 2\,\mathrm{d}\rho_{\mathbf{X}} + \int_{S\setminus(\Omega_\mathcal{S}\cap S)} \mathrm{d}\rho_{\mathbf{X}} + \int_{\Omega_\mathcal{S}\cap S} \mathrm{d}\rho_{\mathbf{X}} \\
&= \int_{(\Omega_\mathcal{S}\setminus S)\cup(S\setminus\Omega_\mathcal{S})} \mathrm{d}\rho_{\mathbf{X}} \\
&= \rho_{\Omega_\mathcal{S}\Delta S}.
\end{aligned} \tag{3.7}$$

Für jedes $k \in \{1,\ldots,n\}$ sei

$$\mathcal{S}_k := \{S \in \mathcal{S} \colon (k-1)\varepsilon_n < \sigma_S^2 \leq k\varepsilon_n\}$$

und

$$\mathcal{S}_0 := \{S \in \mathcal{S} \colon \rho_{S\Delta\Omega_S} = 0\}.$$

Dann gilt schon einmal per Konstruktion $\mathcal{S}_k \subseteq \mathcal{S}$ für alle $k \in \{0,\ldots,n\}$. Also gilt auch

$$\bigcup_{i=0}^{n} \mathcal{S}_k \subseteq \mathcal{S}.$$

Für $S \in \mathcal{S}$ und $\rho_{S\Delta\Omega_S} = 0$ folgt wegen (3.7) die Ungleichung $\sigma_S^2 \leq \rho_{S\Delta\Omega_S} = 0$. Damit ist

$$\{S \in \mathcal{S} \colon \sigma_S^2 = 0\} \subseteq \mathcal{S}_0.$$

Für $S \in \mathcal{S}$ ist $0 \leq \sigma_S^2$ und wegen (3.7) hat man auch $\sigma_S^2 \leq \rho_{\Omega_S \Delta S} \leq 1$. Da

$$n\varepsilon_n = n \cdot \frac{K \ln n}{n} \geq n \cdot \frac{1}{n} = 1 \tag{3.8}$$

für $K \geq 2$ und $n \geq 2$ gilt, existiert per Konstruktion ein $k \in \{0, \ldots, n\}$ mit $S \in \mathcal{S}_k$. Somit gilt aber auch

$$\bigcup_{i=0}^{n} \mathcal{S}_k \supseteq \mathcal{S}$$

und zusammenfassend

$$\bigcup_{i=0}^{n} \mathcal{S}_k = \mathcal{S}.$$

Für $k \in \{0, \ldots, n\}$ definieren wir nun $\mu := \mu_k := \varepsilon_n \sqrt{k}$. Bei $k = 0$ und $S \in \mathcal{S}_0$ folgt dann

$$e_n(S) = \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n \geq \sigma_S \sqrt{\varepsilon_n} + \varepsilon_n = \varepsilon_n \geq 0 = \mu_k$$

und im Fall $k \in \{1, \ldots, n\}$ mit $S \in \mathcal{S}_k$ ergibt sich

$$e_n(S) = \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n > \sqrt{(k-1)\varepsilon_n^2} + \varepsilon_n = (\sqrt{k-1} + 1)\varepsilon_n \geq \varepsilon_n \sqrt{k} = \mu_k.$$

Daraus folgt nun

$$\begin{aligned} &\mathbf{P}\left[\sup_{S \in \mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\right] \\ &\leq \mathbf{P}\left[\sup_{S \in \mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \mu_k\right]. \end{aligned} \tag{3.9}$$

Für $S \in \mathcal{S}_0$ ist $\mu_k = 0$ und $\rho_{S\Delta\Omega_S} = 0$. Wegen $\rho_{S\Delta\Omega_S} = 0$ ist $\mathbf{P}[|(\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > 0] = 0$ und nach (3.5) dann auch $|\eta_{\Omega_S} - \eta_S| = 0$. Also gilt

$$0 = \mathbf{P}\left[\sup_{S \in \mathcal{S}_0} |\eta_{\Omega_S} - \eta_S| + |(\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > 0\right] \geq \mathbf{P}\left[\sup_{S \in \mathcal{S}_0} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > 0\right].$$

Im Folgenden brauchen wir $k = 0$ nicht mehr zu beachten und können von $k \in \{1, \ldots, n\}$ ausgehen. Auf (3.9) wird nun Talagrand's Konzentrationsungleichung für die Menge von Funktionen

$$\mathcal{F}_k := \{f_S - \mathbf{E}[f_S(X, Y)] \colon S \in \mathcal{S}_k\} \tag{3.10}$$

angewendet, siehe Lemma A.2. Man beachte, dass auch $S_k = \emptyset$ gelten kann. Indizes k mit $S_k = \emptyset$ werden bei Anwendung des Lemmas ausgespart, da es für diese nicht definiert ist. Jede Funktion $f \in \mathcal{F}_k$ hat per Konstruktion den Erwartungswert 0 und

Varianz $\sigma_S^2 \leq \rho_{\Omega_S \Delta S}$ nach (3.7). Außerdem gilt $\|f\|_\infty \leq 2$ wegen (3.6). Definiert man nun für $\mathbf{x} \in \mathbf{X}^n$ und $\mathbf{y} \in \mathbf{Y}^n$ die Zufallsvariable

$$\begin{aligned} Z_k(\mathbf{x}, \mathbf{y}) &:= \sup_{S \in \mathcal{S}_k} \left| \sum_{j=1}^{n} \Big(f_S(x_j, y_j) - \mathbf{E}[f_S(X, Y)] \Big) \right| \\ &= n \sup_{S \in \mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| \end{aligned}$$

und

$$\sup_{S \in \mathcal{S}_k} \sigma_S \leq \sqrt{k\varepsilon_n} = \sqrt{\frac{kK \ln n}{n}} =: \sigma_k,$$

so sind die Voraussetzungen von Talagrand's Konzentrationsungleichung (Lemma A.2) erfüllt und man erhält wegen

$$\sigma^2 = n \sup_{S \in \mathcal{S}_k} \sigma_S^2 \leq nk\varepsilon_n,$$

dann

$$\mathbf{P}\left[|Z_k - \mathbf{E}[Z_k]| > t\right] \leq C_0 \exp\left(-\frac{t}{2c_0} \log\left(1 + \frac{2t}{nk\varepsilon_n + 2\mathbf{E}[Z_k]} \right) \right) \tag{3.11}$$

für $t > 0$, wobei C_0, c_0 absolute Konstanten darstellen. Für jedes $t > 2\mathbf{E}[Z_k]$ gilt $t/2 > \mathbf{E}[Z_k]$ und somit

$$\{Z_k > t\} \subseteq \{Z_k - \mathbf{E}[Z_k] > t/2\} \subseteq \{|Z_k - \mathbf{E}[Z_k]| > t/2\}.$$

Für ein solches t ergibt sich aus der Abschätzung der Wahrscheinlichkeit (3.11)

$$\begin{aligned} \mathbf{P}[Z_k > t] &\leq \mathbf{P}[|Z_k - \mathbf{E}[Z_k]| > t/2] \\ &\leq C_0 \cdot \exp\left(-\frac{t}{4c_0} \cdot \ln\left(1 + \frac{t}{nk\varepsilon_n + 2\mathbf{E}[Z_k]} \right) \right). \end{aligned} \tag{3.12}$$

Für $t = n\mu_k = n\varepsilon_n\sqrt{k} \geq 2\mathbf{E}[Z_k]$ folgt daraus

$$\begin{aligned} \mathbf{P}[Z_k > n\mu_k] &\leq C_0 \cdot \exp\left(-\frac{n\mu_k}{4c_0} \cdot \ln\left(1 + \frac{n\mu_k}{nk\varepsilon_n + 2\mathbf{E}[Z_k]} \right) \right) \\ &\leq C_0 \cdot \exp\left(-\frac{n\mu_k}{4c_0} \cdot \ln\left(1 + \frac{n\mu_k}{nk\varepsilon_n + n\varepsilon_n\sqrt{k}} \right) \right) \\ &\leq C_0 \cdot \exp\left(-\frac{n\mu_k}{4c_0} \cdot \ln\left(1 + \frac{\mu_k}{2k\varepsilon_n} \right) \right). \end{aligned} \tag{3.13}$$

Nutzt man nun $\ln(1+x) \geq x/2$ für $x \in [0;1]$ so ist

$$\ln\left(1+\frac{\mu_k}{2k\varepsilon_n}\right) = \ln\left(1+\frac{1}{2\sqrt{k}}\right) \geq \frac{1}{4\sqrt{k}}$$

und aus (3.13) erhält man dann

$$\begin{aligned}
\mathbf{P}\left[\sup_{S\in\mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \mu_k\right] &= \mathbf{P}[Z_k > n\mu_k] \\
&\leq C_0 \cdot \exp\left(-\frac{n\mu_k}{4c_0} \cdot \ln\left(1+\frac{\mu_k}{2k\varepsilon_n}\right)\right) \\
&\leq C_0 \cdot \exp\left(-\frac{n\mu_k}{4c_0} \cdot \frac{1}{4\sqrt{k}}\right) \\
&= C_0 \cdot \exp\left(-\frac{n\varepsilon_n}{16c_0}\right) \qquad (3.14) \\
&= C_0 \cdot \exp\left(-\frac{K\ln n}{16c_0}\right) \\
&\leq C_0 \cdot \exp\left(-(r+1)\ln n\right) \\
&\leq C_0 n^{-r-1}
\end{aligned}$$

für $K \geq 16c_0(r+1)$.

Zum Schluss des Beweises folgt ein Vereinigungsargument. Sei $S' \in \mathcal{S} = \bigcup_{i=0}^{n} \mathcal{S}_k$, so existiert ein $k \in \{0,\ldots,n\}$ mit $S' \in \mathcal{S}_k$. Außerdem gilt dann

$$\begin{aligned}
&\{\exists S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > e_n(S)\} \\
&= \{\exists S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\} \\
&\subseteq \left\{\sup_{S\in\mathcal{S}} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\right\} \\
&\subseteq \bigcup_{k=0}^{n} \left\{\sup_{S\in\mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\right\},
\end{aligned}$$

sowie mit unserer ersten Abschätzung der Wahrscheinlichkeit (3.9) in Kombination mit (3.14) folgt weiter

$$\begin{aligned}
&\mathbf{P}[\exists S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > e_n(S)] \\
&\leq \mathbf{P}\left[\bigcup_{k=0}^{n} \left\{\sup_{S\in\mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\right\}\right] \\
&\leq \sum_{k=0}^{n} \mathbf{P}\left[\sup_{S\in\mathcal{S}_k} |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| - e_n(S) > 0\right]
\end{aligned}$$

$$\leq \sum_{k=0}^{n} \mathbf{P}\left[\sup_{S\in\mathcal{S}_k} |\eta_{\Omega_\mathcal{S}} - \eta_S - (\bar{\eta}_{\Omega_\mathcal{S}} - \bar{\eta}_S)| > \mu_k\right]$$
$$\leq 0 + \sum_{k=1}^{n} C_0 n^{-r-1} = C_0 n^{-r}.$$

Daraus lässt sich nun

$$\begin{aligned}&\mathbf{P}[\forall S \in \mathcal{S}\colon\ |\eta_{\Omega_\mathcal{S}} - \eta_S - (\bar{\eta}_{\Omega_\mathcal{S}} - \bar{\eta}_S)| \leq e_n(S)]\\ &= 1 - \mathbf{P}[\exists S \in \mathcal{S}\colon\ |\eta_{\Omega_\mathcal{S}} - \eta_S - (\bar{\eta}_{\Omega_\mathcal{S}} - \bar{\eta}_S)| > e_n(S)]\\ &\geq 1 - C_0 n^{-r}\end{aligned}$$

schlussfolgern.

Es bleibt noch zu zeigen, dass $t \geq 2\mathbf{E}[Z_k]$ für eine hinreichend große Wahl von K gilt. Wir fordern nun

$$K \geq \frac{C_2^2}{\ln 2}, \tag{3.15a}$$
$$K \geq 128 V_\mathcal{S} C_1^2, \tag{3.15b}$$
$$K \geq \frac{V_\mathcal{S}}{2} \tag{3.15c}$$

mit noch festzulegenden Konstanten $C_1, C_2 > 0$. Als Hilfsergebnis erhalten wir dann

$$\begin{aligned}\ln\left(\frac{C_2^2 n}{K \cdot \ln n}\right) &\overset{(3.15a)}{\leq} \ln\left(\frac{C_2^2 n \cdot \ln 2}{C_2^2 \cdot \ln n}\right)\\ &\overset{n\geq 2}{\leq} \ln\left(\frac{n \cdot \ln n}{\ln n}\right)\\ &= \ln n.\end{aligned}$$

Deswegen ergibt sich

$$\begin{aligned}&\frac{V_\mathcal{S}}{2} \cdot \ln\left(\frac{C_2^2 n}{K \cdot \ln n}\right) \leq \frac{V_\mathcal{S}}{2} \cdot \ln n\\ \Rightarrow\quad & V_\mathcal{S} \ln\left(C_2 \sigma_k^{-1}\right) = \frac{V_\mathcal{S}}{2} \cdot \ln\left(\frac{C_2^2 n}{kK \cdot \ln n}\right) \leq \frac{V_\mathcal{S}}{2} \cdot \ln\left(\frac{C_2^2 n}{K \cdot \ln n}\right) \leq \frac{V_\mathcal{S}}{2} \cdot \ln n\\ & \frac{V_\mathcal{S}}{2} \cdot \ln n \overset{(3.15b)}{\leq} \frac{K}{256 C_1^2} \cdot \ln n \qquad \wedge \qquad \frac{V_\mathcal{S}}{2} \cdot \ln n \overset{(3.15c)}{\leq} K \cdot \ln n\\ \Rightarrow\quad & V_\mathcal{S} \ln\left(C_2 \sigma_k^{-1}\right) \leq \min\left\{\frac{K \cdot \ln n}{256 C_1^2}; K \cdot \ln n\right\}\\ \Rightarrow\quad & \frac{V_\mathcal{S}}{n} \ln\left(C_2 \sigma_k^{-1}\right) \leq \min\left\{\frac{\varepsilon_n}{256 C_1^2}; \varepsilon_n\right\} \leq \min\left\{\frac{\varepsilon_n}{256 C_1^2}; k\varepsilon_n\right\}\\ \Rightarrow\quad & \sqrt{\frac{V_\mathcal{S}}{n} \ln\left(C_2 \sigma_k^{-1}\right)} \leq \min\left\{\frac{\sqrt{\varepsilon_n}}{16 C_1}; \sqrt{k\varepsilon_n}\right\}.\end{aligned}$$

Ferner folgt daraus

$$\sigma_k\sqrt{\frac{V_{\mathcal{S}}}{n}\ln\left(C_2\sigma_k^{-1}\right)} = \sqrt{\varepsilon_n k}\cdot\sqrt{\frac{V_{\mathcal{S}}}{n}\ln\left(C_2\sigma_k^{-1}\right)} \leq \sqrt{\varepsilon_n k}\cdot\frac{\sqrt{\varepsilon_n}}{16C_1} = \frac{\varepsilon_n\sqrt{k}}{16C_1} \tag{3.16}$$

und

$$\sqrt{\frac{V_{\mathcal{S}}}{n}\ln\left(C_2\sigma_k^{-1}\right)} \leq \sqrt{k\varepsilon_n} = \sigma_k. \tag{3.17}$$

Mit (3.17) ergibt sich sofort

$$\max\left\{\sigma_k; \sqrt{\frac{V_{\mathcal{S}}}{n}\ln\left(C_2\sigma_k^{-1}\right)}\right\} = \sigma_k. \tag{3.18}$$

Jetzt schätzen wir $\mathbf{E}[Z_k]$ ab und nutzen dazu bekannte Schranken für den Erwartungswert von Supremumsnormen von empirischen Prozessen, hier Lemma A.12. Mit den dortigen Bezeichnungen wählen wir

$$F = \mathbb{1}_{\mathbf{z}}, \qquad U = 1, \qquad A = C_2, \qquad V = 4V_{\mathcal{S}}, \qquad \sigma = \sigma_k, \qquad C = C_1,$$

dann ergibt sich

$$\mathbf{E}[Z_k] \leq 8n\cdot C_1 \max\left\{\sigma_k, \sqrt{\frac{V_{\mathcal{S}}\ln\left(C_2\sigma_k^{-1}\right)}{n}}\right\}\cdot\sqrt{\frac{V_{\mathcal{S}}\ln\left(C_2\sigma_k^{-1}\right)}{n}}.$$

Nutzt man damit nun auch (3.18) und (3.16), so erhält man

$$\begin{aligned}
2\mathbf{E}[Z_k] &\leq 16n\cdot C_1 \max\left\{\sigma_k, \sqrt{\frac{V_{\mathcal{S}}\ln\left(C_2\sigma_k^{-1}\right)}{n}}\right\}\cdot\sqrt{\frac{V_{\mathcal{S}}\ln\left(C_2\sigma_k^{-1}\right)}{n}} \\
&\overset{(3.18)}{=} 16n\cdot C_1\sigma_k\cdot\sqrt{\frac{V_{\mathcal{S}}\ln\left(C_2\sigma_k^{-1}\right)}{n}} \\
&\overset{(3.16)}{\leq} 16n\cdot C_1\cdot\frac{\varepsilon_n\sqrt{k}}{16C_1} \\
&= n\varepsilon_n\sqrt{k} = n\mu_k = t.
\end{aligned}$$

Die Forderungen an K sind wegen (3.8), (3.14) und (3.15) also

$$K \geq 2, \qquad K \geq 16c_0(r+1), \qquad K \geq \frac{C_2^2}{\ln 2}, \qquad K \geq 128V_{\mathcal{S}}C_1^2, \qquad K \geq \frac{V_{\mathcal{S}}}{2}.$$

Es gilt

$$\begin{aligned}&\max\left\{2; 16c_0(r+1); \frac{C_2^2}{\ln 2}; 128V_{\mathcal{S}}C_1^2; \frac{V_{\mathcal{S}}}{2}\right\}\\ &\leq \max\{r+1, V_{\mathcal{S}}\} \cdot \underbrace{2 \cdot \max\left\{1; 8c_0; \frac{C_2^2}{2\ln 2}; 64C_1^2\right\}}_{=:B}.\end{aligned}$$

Somit erfüllt

$$K = B \cdot \max\{r+1, V_{\mathcal{S}}\}$$

alle geforderten Bedingungen. □

Satz 3.2. *Sei $\mathcal{S}$ eine endliche Familie von Mengen. Für jedes $r > 0$ definieren wir*

$$e_n(S) := \sqrt{\rho_{S\Delta\Omega_{\mathcal{S}}} \cdot \varepsilon_n} + \varepsilon_n, \qquad \varepsilon_n := \frac{14(r\ln n + \ln|\mathcal{S}|)}{3 \cdot n}. \tag{3.19}$$

Dann erhalten wir mit Wahrscheinlichkeit mindestens $1 - 2n^{-r}$ auf den Beobachtungsdaten $\mathbf{z} \in \mathbf{Z}^n$ für alle $S \in \mathcal{S}$ die Abschätzung

$$|\eta_S - \eta_{\Omega_{\mathcal{S}}} - (\bar{\eta}_S - \bar{\eta}_{\Omega_{\mathcal{S}}})| \leq e_n(S). \tag{3.20}$$

Beweis. Wir betrachten wie im davor geführten Beweis von Satz 3.1 für $S \in \mathcal{S}$ die Zufallsvariable $f_S(X,Y) = Y\mathbb{1}_{\Omega_{\mathcal{S}}}(X) - Y\mathbb{1}_S(X)$. Dazu haben wir schon einige Ergebnisse zusammengetragen. Für den Erwartungswert dieser Zufallsvariablen gilt nach (3.3):

$$\mathbf{E}\left[f_S(X,Y)\right] = \eta_{\Omega_{\mathcal{S}}} - \eta_S.$$

Wegen (3.6) erhält man

$$\left\|f_S(X,Y) - \mathbf{E}\left[f_S(X,Y)\right]\right\|_\infty \leq 2$$

und aus (3.7) ergibt sich für die Varianz

$$\begin{aligned}&\mathbf{Var}\left[f_S(X,Y) - \mathbf{E}\left[f_S(X,Y)\right]\right]\\ &= \mathbf{E}\left[\left(f_S(X,Y) - \mathbf{E}\left[f_S(X,Y)\right]\right)^2\right] - \left(\mathbf{E}\left[f_S(X,Y) - \mathbf{E}\left[f_S(X,Y)\right]\right]\right)^2\\ &= \mathbf{Var}\left[f_S(X,Y)\right] - \left(\mathbf{E}\left[f_S(X,Y)\right] - \mathbf{E}\left[f_S(X,Y)\right]\right)^2\\ &\leq \mathbf{Var}\left[f_S(X,Y)\right]\\ &\leq \rho_{\Omega_{\mathcal{S}}\Delta S}.\end{aligned} \tag{3.21}$$

Damit lässt sich nun die Ungleichung von Bernstein auf die durch unsere Beobachtungsdaten gebildeten Zufallsvariablen

$$X_i := f_S(x_i, y_i) - \mathbf{E}\left[f_S(x_i, y_i)\right], \qquad i \in \{1, \ldots, n\},$$

mit

$$\sigma^2 := \frac{1}{n}\sum_{k=1}^{n} \mathbf{Var}\left[X_i\right] \overset{(3.21)}{\leq} \frac{1}{n}\sum_{k=1}^{n} \rho_{\Omega_S \Delta S} = \rho_{\Omega_S \Delta S} \tag{3.22}$$

anwenden, siehe dazu Lemma A.15. Es gilt also für $\delta > 0$

$$\mathbf{P}\left[\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S) > \delta\right] \overset{(A.10)}{\leq} \exp\left(-\frac{(n\delta)^2}{2n \cdot \sigma^2 + 2 \cdot 2 \cdot n\delta/3}\right) \tag{3.23}$$

und

$$\begin{aligned}\mathbf{P}\left[\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S) < -\delta\right] &= \mathbf{P}\left[-\left(\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)\right) > \delta\right] \\ &\overset{(A.10)}{\leq} \exp\left(-\frac{(n\delta)^2}{2n \cdot \sigma^2 + 2 \cdot 2 \cdot n\delta/3}\right).\end{aligned} \tag{3.24}$$

Kombiniert man die beiden Ungleichungen (3.23), (3.24) und nutzt dann die Abschätzung der Varianz (3.22), so erhält man

$$\begin{aligned}\mathbf{P}\left(\left|\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)\right| > \delta\right) &\leq 2 \cdot \exp\left(-\frac{(n\delta)^2}{2n \cdot \sigma^2 + 2 \cdot 2 \cdot n\delta/3}\right) \\ &\leq 2 \cdot \exp\left(-\frac{n\delta^2}{2\rho_{\Omega_S \Delta S} + 4\delta/3}\right).\end{aligned} \tag{3.25}$$

Für $\delta := e_n(S) := \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n$ mit $\varepsilon_n := \frac{14(r \ln n + \ln|\mathcal{S}|)}{3 \cdot n}$ betrachtet man nun zwei Fälle:

- Im Fall $\varepsilon_n \leq \rho_{S\Delta\Omega_S}$ ist der Zähler der Exponentialfunktion aus (3.25)

  $$n \cdot \left(\sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right)^2 \geq n \cdot \left(\sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n}\right)^2 = n \cdot \rho_{S\Delta\Omega_S} \cdot \varepsilon_n,$$

  sowie der Nenner

  $$\begin{aligned}2\rho_{S\Delta\Omega_S} + 4 \cdot \left(\sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right)/3 &\leq 2\rho_{S\Delta\Omega_S} + \frac{4}{3} \cdot \left(\sqrt{\rho^2_{S\Delta\Omega_S}} + \rho_{S\Delta\Omega_S}\right) \\ &= \frac{14}{3} \cdot \rho_{S\Delta\Omega_S}.\end{aligned}$$

- Im anderen Fall $\varepsilon_n > \rho_{S\Delta\Omega_S}$ ergibt sich für den Zähler

$$n \cdot \left(\sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right)^2 \geq n \cdot \varepsilon_n^2$$

und für den Nenner

$$2\rho_{S\Delta\Omega_S} + 4 \cdot \left(\sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right)/3 \leq 2\varepsilon_n + \frac{4}{3} \cdot \left(\sqrt{\varepsilon_n^2} + \varepsilon_n\right) = \frac{14}{3} \cdot \varepsilon_n.$$

Daraus folgt in beiden Fällen

$$-\frac{n\delta^2}{2\rho_{\Omega_S\Delta S} + 4\delta/3} \leq -\frac{3n \cdot \varepsilon_n}{14}.$$

Also ergibt sich

$$\begin{aligned}\mathbf{P}\left(|\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right) &\leq 2 \cdot \exp\left(-\frac{3n \cdot \varepsilon_n}{14}\right) \\ &= 2\left(|\mathcal{S}|^{-1} n^{-r}\right)\end{aligned}$$

und schließlich

$$\begin{aligned}&\mathbf{P}\left(\forall S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| \leq e_n(S)\right) \\ &= \mathbf{P}\left(\forall S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| \leq \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right) \\ &= 1 - \mathbf{P}\left(\exists S \in \mathcal{S}\colon\ |\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right) \\ &= 1 - \mathbf{P}\left(\bigcup_{S\in\mathcal{S}} \left\{|\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right\}\right) \\ &\geq 1 - \sum_{S\in\mathcal{S}} \mathbf{P}\left(|\eta_{\Omega_S} - \eta_S - (\bar{\eta}_{\Omega_S} - \bar{\eta}_S)| > \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n\right) \\ &\geq 1 - \sum_{S\in\mathcal{S}} 2\left(|\mathcal{S}|^{-1} n^{-r}\right) \\ &= 1 - 2n^{-r}.\end{aligned}$$

□

Um den Schätzfehler bzgl. einer Klasse $\mathcal{S}$ zu analysieren, definieren wir den folgenden Modulus:

Definition 3.3.

$$\omega(\rho, e_n, \mathcal{S}) := \sup\left\{ \int_{S\Delta\Omega_S} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \;\middle|\; S \in \mathcal{S};\ \int_{S\Delta\Omega_S} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \leq 3e_n(S) \right\}. \tag{3.26}$$

Dabei ist $e_n\colon \mathcal{S} \to \mathbb{R}$ eine gegebene Mengenfunktion für $\mathcal{S} \subseteq \{S\colon S \subseteq \mathbf{X}\}$. Deswegen schreiben wir auch $\omega(\rho, e_n)$ für $\omega(\rho, e_n, \mathcal{S})$.

Um zu kennzeichnen, dass die folgenden Resultate auch für von $\bar{\eta}_S$ verschiedene Schätzer genutzt werden können, verwenden wir im Folgenden die allgemeinere Form $\hat{\eta}_S$ für eine Menge $S \subseteq \mathbf{X}$, vgl. dazu den Abschnitt von (2.4) und (2.5).

Satz 3.4. *Gelte für jedes $S \in \mathcal{S}$ die Bedingung*

$$|\eta_S - \eta_{\Omega_\mathcal{S}} - (\hat{\eta}_S - \hat{\eta}_{\Omega_\mathcal{S}})| \leq e_n(S) \tag{3.27}$$

für Beobachtungsdaten $\mathbf{z}$. Dann gilt für die gleiche Beobachtung

$$R(\hat{\Omega}_\mathcal{S}) - R(\Omega_\mathcal{S}) \leq \max\{\omega(\rho, e_n), a(\Omega^*, \mathcal{S})\}. \tag{3.28}$$

Beweis. Wir betrachten Beobachtungsdaten $\mathbf{z}$, für die (3.27) erfüllt ist und zeigen, dass dann auch (3.28) für diese $\mathbf{z}$ gilt. Sei $S_0 := \Omega_\mathcal{S} \backslash \hat{\Omega}_\mathcal{S}$ und $S_1 := \hat{\Omega}_\mathcal{S} \backslash \Omega_\mathcal{S}$, dann gilt per Konstruktion $S_0 \cup S_1 = \hat{\Omega}_\mathcal{S} \Delta \Omega_\mathcal{S}$ und $S_0 \cap S_1 = \emptyset$. Man beachte, dass im Gegensatz zu $\hat{\Omega}_\mathcal{S}$ und $\Omega_\mathcal{S}$ die Mengen S_0, S_1 im Allgemeinen nicht in $\mathcal{S}$ liegen müssen. Wir beginnen mit der Gleichung

$$\begin{aligned}
&R(\hat{\Omega}_\mathcal{S}) - R(\Omega_\mathcal{S}) \\
&\overset{(2.2)}{=} \int_{\mathbf{X}} p \, \mathrm{d}\rho_\mathbf{X} - \int_{\hat{\Omega}_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} - \int_{\mathbf{X}} p \, \mathrm{d}\rho_\mathbf{X} + \int_{\Omega_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} \\
&\overset{(2.1)}{=} \eta_{\Omega_\mathcal{S}} - \eta_{\hat{\Omega}_\mathcal{S}} \\
&= \int_{\Omega_\mathcal{S} \backslash \hat{\Omega}_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} + \int_{\Omega_\mathcal{S} \cap \hat{\Omega}_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} - \left(\int_{\hat{\Omega}_\mathcal{S} \backslash \Omega_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} + \int_{\Omega_\mathcal{S} \cap \hat{\Omega}_\mathcal{S}} \eta \, \mathrm{d}\rho_\mathbf{X} \right) \\
&= \eta_{S_0} - \eta_{S_1}.
\end{aligned} \tag{3.29}$$

Wir können nun annehmen, dass $\eta_{S_0} - \eta_{S_1} > 0$ gilt, denn andernfalls braucht man nichts zu beweisen. Aus der Definition von $\hat{\Omega}_\mathcal{S}$ und $\Omega_\mathcal{S}$ wissen wir, dass

$$\hat{\eta}_{\hat{\Omega}_\mathcal{S}} - \hat{\eta}_{\Omega_\mathcal{S}} \geq 0$$

ist. In Verbindung mit (3.29) und (3.27) ergibt das

$$\begin{aligned}
\eta_{S_0} - \eta_{S_1} &\overset{(3.29)}{=} \eta_{\Omega_\mathcal{S}} - \eta_{\hat{\Omega}_\mathcal{S}} \\
&\leq |\eta_{\Omega_\mathcal{S}} - \eta_{\hat{\Omega}_\mathcal{S}} + (\hat{\eta}_{\hat{\Omega}_\mathcal{S}} - \hat{\eta}_{\Omega_\mathcal{S}})| \\
&= |\eta_{\hat{\Omega}_\mathcal{S}} - \eta_{\Omega_\mathcal{S}} - (\hat{\eta}_{\hat{\Omega}_\mathcal{S}} - \hat{\eta}_{\Omega_\mathcal{S}})| \\
&\overset{(3.27)}{\leq} e_n(\hat{\Omega}_\mathcal{S}).
\end{aligned} \tag{3.30}$$

Ferner definieren wir für eine Menge $S \subseteq \mathbf{X}$, die Menge $S^+ := S \cap \Omega^*$ und $S^- := S \cap (\Omega^*)^c$. Nach Definition von Ω^* ist $\eta \geq 0$ in S^+ und $\eta < 0$ in S^-. Weiter hat man $S = S^+ \cup S^-$ und $S^+ \cap S^- = \emptyset$. Somit kann man Folgendes schreiben:

$$\eta_{S_0} - \eta_{S_1} = A - B, \qquad A := \eta_{S_0^+} - \eta_{S_1^-}, \qquad B := \eta_{S_1^+} - \eta_{S_0^-}. \tag{3.31}$$

Dabei sind $A, B \geq 0$. Wir unterscheiden nun zwei Fälle:

- $A \leq 2B$: Wegen $S_0^- \subseteq \Omega_{\mathcal{S}} \backslash \Omega^*$ und $\eta < 0$ außerhalb von Ω^* gilt

  $$-\eta_{S_0^-} \leq -\eta_{\Omega_{\mathcal{S}} \backslash \Omega^*}.$$

  Ähnlich dazu folgt wegen $S_1^+ \subseteq \Omega^* \backslash \Omega_{\mathcal{S}}$, sowie $\eta \geq 0$ in Ω^* dann

  $$\eta_{S_1^+} \leq \eta_{\Omega^* \backslash \Omega_{\mathcal{S}}}.$$

  Insgesamt erhält man damit

  $$\begin{aligned} R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega_{\mathcal{S}}) &\overset{(3.29)}{=} \eta_{S_0} - \eta_{S_1} \overset{(3.31)}{=} A - B \leq B = \eta_{S_1^+} - \eta_{S_0^-} \\ &\leq \eta_{\Omega^* \backslash \Omega_{\mathcal{S}}} - \eta_{\Omega_{\mathcal{S}} \backslash \Omega^*} = \eta_{\Omega^*} - \eta_{\Omega_{\mathcal{S}}} \underset{(2.2)}{\overset{(2.1)}{=}} R(\Omega_{\mathcal{S}}) - R(\Omega^*) \\ &= a(\Omega^*, \mathcal{S}). \end{aligned}$$

- $A > 2B$: Zuerst formen wir ein Integral um, welches mit dem Modulus ω in Verbindung steht:

  $$\begin{aligned} \int_{\hat{\Omega}_{\mathcal{S}} \Delta \Omega_{\mathcal{S}}} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} &= \int_{S_0 \cup S_1} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_{S_0} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S_1} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_{S_0^+} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S_0^-} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S_1^+} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S_1^-} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \int_{S_0^+} \eta \, \mathrm{d}\rho_{\mathbf{X}} - \int_{S_0^-} \eta \, \mathrm{d}\rho_{\mathbf{X}} + \int_{S_1^+} \eta \, \mathrm{d}\rho_{\mathbf{X}} - \int_{S_1^-} \eta \, \mathrm{d}\rho_{\mathbf{X}} \\ &= \eta_{S_0^+} - \eta_{S_0^-} + \eta_{S_1^+} - \eta_{S_0^-} \\ &= A + B. \end{aligned}$$

Daraus folgt nun

$$\int_{\hat{\Omega}_{\mathcal{S}} \Delta \Omega_{\mathcal{S}}} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} = A + B < A - 3B + 4B < 3(A - B) \overset{(3.31)}{=} 3(\eta_{S_0} - \eta_{S_1}) \overset{(3.30)}{\leq} 3e_n(\hat{\Omega}_{\mathcal{S}}).$$

Das bedeutet, dass $\hat{\Omega}_{\mathcal{S}}$ eine der Mengen ist, die in der Definition von $\omega(\rho, e_n)$ auftauchen. Daraus ergibt sich

$$R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega_{\mathcal{S}}) = \eta_{S_0} - \eta_{S_1} = A - B \leq \int_{\hat{\Omega}_{\mathcal{S}} \Delta \Omega_{\mathcal{S}}} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \leq \omega(\rho, e_n).$$

Fasst man die beiden Fälle zusammen, so führt das zu

$$R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega_{\mathcal{S}}) \leq \max\{\omega(\rho, e_n), a(\Omega^*, \mathcal{S})\}. \qquad \square$$

Folgerung 3.5. *Gelte für jedes $S \in \mathcal{S}$ die Bedingung* (3.27)

$$|\eta_S - \eta_{\Omega_{\mathcal{S}}} - (\hat{\eta}_S - \hat{\eta}_{\Omega_{\mathcal{S}}})| \leq e_n(S)$$

für Beobachtungsdaten $\mathbf{z}$*. Dann gilt mit den gleichen Beobachtungsdaten*

$$R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega^*) \leq \omega(\rho, e_n) + 2a(\Omega^*, \mathcal{S}).$$

Beweis.

$$\begin{aligned} R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega^*) &= R(\hat{\Omega}_{\mathcal{S}}) - R(\Omega_{\mathcal{S}}) + R(\Omega_{\mathcal{S}}) - R(\Omega^*) \\ &\leq \max\{\omega(\rho, e_n), a(\Omega^*, \mathcal{S})\} + a(\Omega^*, \mathcal{S}) \\ &\leq \omega(\rho, e_n) + 2a(\Omega^*, \mathcal{S}). \end{aligned} \qquad \square$$

3.2 Abschätzung des Modulus durch Tsybakov-Bedingungen

Der im vorherigen Abschnitt eingeführte Modulus ω ist nur schwer durchschaubar und hängt von der Mengenfunktion e_n ab. Wir werden jetzt zeigen, dass für die Typen von Mengenfunktionen e_n, die naturgemäß auftreten, der Modulus eng im Zusammenhang mit den nachfolgenden Bedingungen steht. Annahmen solcher Bedingungen sind ein Hauptbestandteil im Erhalten von Schätzungen über die Leis-

tungsfähigkeit eines empirischen Klassifizierers.

In der folgenden Bedingung (auch Tsybakov-Bedingung genannt) fordert man, dass es ein $\beta \in (0;1)$ gibt, sodass für jedes γ mit $0 < \gamma < \beta$ und für jede $\rho_{\mathbf{X}}$-messbare Menge S die Ungleichung

$$\rho_S \leq C_\gamma \left(\int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \right)^\gamma \tag{3.32}$$

gilt. Dabei ist $C_\gamma > 0$ eine von S unabhängige Konstante.

Lemma 3.6. *Die Bedingung (3.32) ist äquivalent dazu, dass für alle δ mit $0 < \delta < \alpha := \frac{\beta}{1-\beta}$ die Ungleichung*

$$\rho_{\mathbf{X}}(\{x \in \mathbf{X} \colon |\eta(x)| \leq t\}) \leq \bar{C}_\delta t^\delta, \quad 0 < t \leq 1, \tag{3.33}$$

mit einer Konstanten $\bar{C}_\delta > 0$ gilt.

Beweis. Wir nehmen zuerst an, dass die Tsybakov-Bedingung (3.32) gilt. Wir definieren $S_t := \{x \in \mathbf{X} \colon |\eta(x)| \leq t\}$ zu jedem $t \in (0;1]$. Dann gilt (3.32) auch für S_t:

$$\rho_{S_t} \leq C_\gamma \left(\int_{S_t} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \right)^\gamma \leq C_\gamma \left(\int_{S_t} t \, \mathrm{d}\rho_{\mathbf{X}} \right)^\gamma = C_\gamma \left(t \cdot \rho_{S_t} \right)^\gamma = C_\gamma t^\gamma \rho_{S_t}^\gamma,$$

$$\Rightarrow \qquad \rho_{S_t}^{1-\gamma} \leq C_\gamma t^\gamma,$$

$$\Rightarrow \qquad \rho_{S_t} \leq C_\gamma^{\frac{1}{1-\gamma}} \cdot t^{\frac{\gamma}{1-\gamma}} = \bar{C}_\delta \cdot t^\delta,$$

mit $\bar{C}_\delta := C_\gamma^{\frac{1}{1-\gamma}}$ und $\delta = \frac{\gamma}{1-\gamma}$. Außerdem gilt

$$\gamma < \beta \;\Leftrightarrow\; \frac{1}{\beta} < \frac{1}{\gamma} \;\Leftrightarrow\; \frac{1}{\beta} - 1 < \frac{1}{\gamma} - 1 \;\Leftrightarrow\; \frac{\gamma}{1-\gamma} = \frac{1}{\frac{1}{\gamma} - 1} < \frac{1}{\frac{1}{\beta} - 1} = \frac{\beta}{1-\beta}$$

und $\gamma = \frac{\delta}{1+\delta}$. Daraus folgt nun, dass für beliebiges $\delta < \alpha$, das zugehörige γ auch $\gamma < \beta$ erfüllt und schließlich

$$\rho_{\mathbf{X}}(\{x \in \mathbf{X} \colon |\eta(x)| \leq t\}) = \rho_{S_t} \leq \bar{C}_\delta \cdot t^\delta.$$

Gelte nun umgekehrt die Bedingung (3.33). Wir definieren für eine Menge $S \subseteq \mathbf{X}$:

$$S_{\eta=0} := \{x \in S \colon \eta(x) = 0\}.$$

Zuerst zeigen wir, dass für eine messbare Menge $S \subseteq \mathbf{X}$, das Folgende gilt:

$$\rho_{\mathbf{X}}(S_{\eta=0}) = 0. \tag{3.34}$$

Dazu nehmen wir an, dass

$$\rho_{\mathbf{X}}(\mathbf{X}_{\eta=0}) =: c > 0.$$

Da $\bar{C}_\delta t^\delta$ in Abhängigkeit von t eine monoton wachsende, stetige Funktion in $[0, \infty)$ darstellt und $\bar{C}_\delta \cdot 0^\delta = 0$ gilt, so existiert ein $\tilde{t} \in (0, 1]$ mit $0 \leq \bar{C}_\delta \tilde{t}^\delta < c$. Daraus folgt durch Anwendung von (3.33)

$$c = \rho_{\mathbf{X}}(\mathbf{X}_{\eta=0}) \leq \rho_{\mathbf{X}}(\{x \in \mathbf{X} \colon |\eta(x)| \leq \tilde{t}\}) \leq \bar{C}_\delta \tilde{t}^\delta < c.$$

Das ist ein Widerspruch, also war unsere Annahme falsch. Da $\rho_{\mathbf{X}}$ ein Maß ist, bleibt nur der Fall

$$\rho_{\mathbf{X}}(\mathbf{X}_{\eta=0}) = 0$$

übrig. Sei $S \subseteq \mathbf{X}$ eine messbare Menge. Wegen

$$\{x \in S \colon \eta(x) = 0\} \subseteq \{x \in \mathbf{X} \colon \eta(x) = 0\}$$

ergibt sich schließlich

$$\rho_{\mathbf{X}}(\{x \in S \colon \eta(x) = 0\}) \leq \rho_{\mathbf{X}}(\{x \in \mathbf{X} \colon \eta(x) = 0\}) = 0.$$

Nun widmen wir uns der Ungleichung (3.32) und verwenden neben dem gerade Gezeigten (3.34), die Hölder'sche Ungleichung (A.11) (siehe Lemma A.17) mit $p = \frac{1}{\gamma}$ und $q = \frac{1}{1-\gamma}$:

$$\begin{aligned}
\rho_S &= \int_S \mathrm{d}\rho_{\mathbf{X}} = \int_{S_{\eta=0}} \mathrm{d}\rho_{\mathbf{X}} + \int_{S \setminus S_{\eta=0}} \mathrm{d}\rho_{\mathbf{X}} \overset{(3.34)}{=} \int_{S \setminus S_{\eta=0}} 1 \, \mathrm{d}\rho_{\mathbf{X}} = \int_{S \setminus S_{\eta=0}} \frac{|\eta|^\gamma}{|\eta|^\gamma} \, \mathrm{d}\rho_{\mathbf{X}} \\
&\overset{(A.11)}{\leq} \left(\int_{S \setminus S_{\eta=0}} |\eta|^{p \cdot \gamma} \, \mathrm{d}\rho_{\mathbf{X}} \right)^{1/p} \cdot \left(\int_{S \setminus S_{\eta=0}} \frac{1}{|\eta|^{q \cdot \gamma}} \, \mathrm{d}\rho_{\mathbf{X}} \right)^{1/q} \\
&= \left(\int_{S \setminus S_{\eta=0}} |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \right)^{\gamma} \cdot \left(\int_{S \setminus S_{\eta=0}} |\eta|^{-\delta} \, \mathrm{d}\rho_{\mathbf{X}} \right)^{1-\gamma} \\
&\overset{(3.34)}{=} \left(\int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \right)^{\gamma} \cdot \left(\int_{S \setminus S_{\eta=0}} |\eta|^{-\delta} \, \mathrm{d}\rho_{\mathbf{X}} \right)^{1-\gamma}.
\end{aligned}$$

Kann man nun noch zeigen, dass es eine Konstante $C_\gamma > 0$ unabhängig von S gibt, die

$$\left(\int\limits_{S \setminus S_{\eta=0}} |\eta|^{-\delta} \, \mathrm{d}\rho_{\mathbf{X}} \right)^{1-\gamma} \leq C_\gamma \tag{3.35}$$

erfüllt, so ist alles bewiesen. Aus (3.34) erhält man $\rho_{\mathbf{X}}(\mathbf{X}_{\eta=0}) = 0$ und mit Lemma A.18 für $f = |\eta|^{-\delta}$ ergibt sich

$$\begin{aligned}
\int\limits_{S \setminus S_{\eta=0}} |\eta|^{-\delta} \, \mathrm{d}\rho_{\mathbf{X}} &= \int\limits_{\mathbf{X}} \mathbb{1}_{S \setminus S_{\eta=0}} \cdot |\eta|^{-\delta} \, \mathrm{d}\rho_{\mathbf{X}} \overset{(A.12)}{=} \int\limits_0^\infty \rho_{\mathbf{X}}(\{\mathbb{1}_{S \setminus S_{\eta=0}} \cdot |\eta|^{-\delta} \geq t\}) \, \mathrm{d}t \\
&= \int\limits_0^\infty \rho_{\mathbf{X}}\Big(\big\{ |\eta| \leq t^{-1/\delta} \cdot \mathbb{1}_{S \setminus S_{\eta=0}} \big\} \Big) \, \mathrm{d}t \\
&\leq \int\limits_0^\infty \rho_{\mathbf{X}}\Big(\big\{ |\eta| \leq t^{-1/\delta} \big\} \Big) \, \mathrm{d}t \\
&= \int\limits_0^1 \rho_{\mathbf{X}}\Big(\big\{ |\eta| \leq t^{-1/\delta} \big\} \Big) \, \mathrm{d}t + \int\limits_1^\infty \rho_{\mathbf{X}}\Big(\big\{ |\eta| \leq t^{-1/\delta} \big\} \Big) \, \mathrm{d}t \\
&\leq 1 + \int\limits_1^\infty \rho_{\mathbf{X}}\Big(\big\{ |\eta| \leq t^{-1/\delta} \big\} \Big) \, \mathrm{d}t \\
&\overset{(3.33)}{\leq} 1 + \int\limits_1^\infty \bar{C}_{\delta+\varepsilon} t^{-(\delta+\epsilon)/\delta} \, \mathrm{d}t \\
&= 1 + \bar{C}_{\delta+\varepsilon} \int\limits_1^\infty t^{-1-\epsilon/\delta} \, \mathrm{d}t \\
&= 1 + \bar{C}_{\delta+\varepsilon} \cdot \frac{\varepsilon}{\delta} \cdot \int\limits_1^\infty \left(\frac{\mathrm{d}}{\mathrm{d}t} \left(-t^{-\epsilon/\delta} \right) \right) \mathrm{d}t \\
&\overset{\varepsilon<\delta}{\leq} 1 + \bar{C}_{\delta+\varepsilon} \int\limits_1^\infty \left(\frac{\mathrm{d}}{\mathrm{d}t} \left(-t^{-\epsilon/\delta} \right) \right) \mathrm{d}t \\
&= 1 - \bar{C}_{\delta+\varepsilon} \left(\lim_{b \to \infty} \left(\frac{1}{b} \right)^{\epsilon/\delta} - \left(\frac{1}{1} \right)^{\epsilon/\delta} \right) \\
&= 1 + \bar{C}_{\delta+\varepsilon}.
\end{aligned}$$

Dabei war $\epsilon > 0$ so klein gewählt, dass $\delta + \epsilon < \alpha$ bei der Anwendung von (3.33) gilt und ebenso $\varepsilon < \delta$. Mit der Wahl $C_\gamma := \left(1 + \bar{C}_{\delta+\varepsilon}\right)^{1-\gamma}$ ist (3.35) erfüllt und somit alles gezeigt. □

Wir wollen nun die Verbindung zwischen dem Modulus ω und der Bedingung (3.32) hervorheben. In der Definition von ω und der Anwendung auf Abschätzungen der Varianz, nehmen wir an, dass wir einen Schätzer besitzen, für den (3.27) mit Wahrscheinlichkeit $1-\tau$ gilt. Dabei wird jedoch nur angenommen, dass das für Mengen $S \in \mathcal{S}$ erfüllt ist, im Unterschied zu (3.32). Der Vergleich erfolgt, wenn die Mengenfunktion e_n wie in den Ergebnissen des vorherigen Kapitels von der Form

$$e_n(S) = \sqrt{\varepsilon_n \cdot \rho_{S\Delta\Omega_S}} + \varepsilon_n$$

ist.

Definition 3.7. Wir definieren die Funktion

$$\phi(\rho, t) := \sup \left\{ \int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} : \int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \leq 3(t + \sqrt{t \cdot \rho_S}) \right\}. \tag{3.36}$$

In dieser Definition sind alle $\rho_{\mathbf{X}}$-messbaren Mengen S erlaubt (nicht notwendigerweise aus $\mathcal{S}$). Unter der Annahme an die Form von e_n ergibt (3.26) dann

$$\omega(\rho, e_n) \leq \phi(\rho, \varepsilon_n).$$

Somit liefert uns das Abklingen von ϕ eine obere Schranke für das Abklingen von ω.

Definition 3.8. Wir sagen ρ erfüllt die ϕ-Bedingung der Ordnung $u > 0$, falls

$$\phi(\rho, t) \leq C_0 \cdot t^u, \qquad 0 < t \leq 1, \tag{3.37}$$

für Konstanten $C_0 > 0$ und $u > 0$ gilt.

Satz 3.9. *Sei ρ ein Maß, welches der Tsybakov-Bedingung (3.32) für einen gegebenen Wert $\beta \in (0;1)$ genügt. Dann erfüllt ρ für jedes γ mit $0 < \gamma < \beta$ auch die ϕ-Bedingung (3.37) für $u = \frac{1}{2-\gamma}$ mit einer Konstanten $C_0 > 0$, die nur von γ und der zugehörigen Konstanten C_γ abhängt. Umgekehrt gilt, falls ρ die ϕ-Bedingung (3.37) mit $u = \frac{1}{2-\gamma}$ für jedes $\gamma \in (0;\beta)$ und zugehöriger Konstante $C_0 > 0$ erfüllt, so genügt ρ auch (3.32) der Ordnung β, wobei die Konstanten $C_\gamma > 0$ nur von u und C_0 abhängen.*

Beweis. Angenommen ρ erfüllt (3.32) für einen Wert β. Sei $\gamma \in (0;\beta)$ und C_γ die zugehörige Konstante. Um die ϕ-Bedingung für $u = \frac{1}{2-\gamma}$ zu überprüfen, sei $t \in (0;1]$

und S so gewählt, dass folgendes gilt:

$$\int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \leq 3 \left(t + \sqrt{t \cdot \rho_S} \right). \tag{3.38}$$

(3.32) liefert dann

$$\rho_S \leq C_\gamma \left(\int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} \right)^\gamma \leq C_\gamma \cdot 3^\gamma \left(\sqrt{t \cdot \rho_S} + t \right)^\gamma. \tag{3.39}$$

Daraus lässt sich ableiten, dass eine Konstante M existiert, für die Folgendes gilt

$$\rho_S \leq M \cdot t^{\frac{\gamma}{2-\gamma}}, \tag{3.40}$$

wobei M nur von C_γ und γ abhängt. Um das zu beweisen, nehmen wir das Gegenteil an. Für alle $M > 0$ gilt

$$\rho_S > M \cdot t^{\frac{\gamma}{2-\gamma}}. \tag{3.41}$$

Beim Umformen von (3.39) erhält man Schrittweise

$$\begin{aligned}
& \rho_S \leq C_\gamma \cdot 3^\gamma \left(\sqrt{t \cdot \rho_S} + t \right)^\gamma, \\
\Rightarrow \quad & \rho_S^{\frac{1}{\gamma}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} \cdot \rho_S^{\frac{1}{2}} + t \right), \\
\Rightarrow \quad & \rho_S^{\frac{1}{\gamma}} \cdot \rho_S^{-\frac{1}{2}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t \cdot \rho_S^{-\frac{1}{2}} \right), \\
\Rightarrow \quad & \rho_S^{\frac{2-\gamma}{2\gamma}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t \cdot \rho_S^{-\frac{1}{2}} \right).
\end{aligned}$$

Benutzt man nun (3.41) zur Abschätzung von ρ_S für beide Seiten, so erhält man daraus

$$\begin{aligned}
& M^{\frac{2-\gamma}{2\gamma}} \cdot t^{\frac{2-\gamma}{2\gamma} \cdot \frac{\gamma}{2-\gamma}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t \cdot M^{-\frac{1}{2}} \cdot t^{-\frac{1}{2} \cdot \frac{\gamma}{2-\gamma}} \right), \\
\Rightarrow \quad & M^{\frac{2-\gamma}{2\gamma}} \cdot t^{\frac{1}{2}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t^{\frac{4-2\gamma}{4-2\gamma}} \cdot M^{-\frac{1}{2}} \cdot t^{-\frac{1}{2} \cdot \frac{\gamma}{2-\gamma}} \right), \\
\Rightarrow \quad & M^{\frac{2-\gamma}{2\gamma}} \cdot t^{\frac{1}{2}} \leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t^{\frac{4-3\gamma}{4-2\gamma}} \cdot M^{-\frac{1}{2}} \right).
\end{aligned} \tag{3.42}$$

Da $\gamma \in (0; \beta) \subseteq (0; 1)$ liegt, ist $4 - 2\gamma > 0$ und es folgt

$$\gamma \leq 1 \quad \Rightarrow \quad 2\gamma \leq 2 \quad \Rightarrow \quad 2 - \gamma \leq 4 - 3\gamma \quad \Rightarrow \quad \frac{1}{2} \leq \frac{4 - 3\gamma}{4 - 2\gamma}.$$

Wendet man das auf (3.42) an, so ergibt sich wegen $t \in (0;1]$ dann

$$t^{\frac{4-3\gamma}{4-2\gamma}} \leq t^{\frac{1}{2}}$$

und somit

$$\begin{aligned} M^{\frac{2-\gamma}{2\gamma}} \cdot t^{\frac{1}{2}} &\leq C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t^{\frac{1}{2}} \cdot M^{-\frac{1}{2}}\right), \\ \Rightarrow \quad t^{\frac{1}{2}} &\leq M^{-\frac{2-\gamma}{2\gamma}} C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(t^{\frac{1}{2}} + t^{\frac{1}{2}} \cdot M^{-\frac{1}{2}}\right), \\ \Rightarrow \quad 1 &\leq M^{-\frac{2-\gamma}{2\gamma}} \cdot C_\gamma^{\frac{1}{\gamma}} \cdot 3 \cdot \left(1 + M^{-\frac{1}{2}}\right). \end{aligned}$$

Wählt man M in Abhängigkeit von C_γ und γ groß genug, so wird die rechte Seite kleiner 1. Damit erhält man einen Widerspruch und das beweist (3.40).

Wegen $\gamma \in (0;\beta) \subseteq (0;1)$ ist

$$\frac{1}{2} + \frac{1}{2} \cdot \frac{\gamma}{2-\gamma} = \frac{2-\gamma+\gamma}{2(2-\gamma)} = \frac{1}{2-\gamma}.$$

Nach unserer Wahl von S hat man die Eigenschaft (3.38). Wendet man darauf (3.40) an, so folgt:

$$\begin{aligned} \int_S |\eta| \, \mathrm{d}\rho_{\mathbf{X}} &\leq 3\left(t + \sqrt{t \cdot \rho_S}\right) \\ &\leq 3\left(t + t^{\frac{1}{2}} \cdot M^{\frac{1}{2}} \cdot t^{\frac{1}{2} \cdot \frac{\gamma}{2-\gamma}}\right) \\ &= 3\left(t + M^{\frac{1}{2}} t^{\frac{1}{2-\gamma}}\right) \\ &= 3\left(t^{\frac{1-\gamma}{2-\gamma}} + M^{\frac{1}{2}}\right) t^{\frac{1}{2-\gamma}} \\ &\leq 3\left(1 + M^{\frac{1}{2}}\right) t^{\frac{1}{2-\gamma}} = C_0 \cdot t^u \end{aligned}$$

mit $u = \frac{1}{2-\gamma}$ und $C_0 = 3\left(1 + M^{\frac{1}{2}}\right)$. Damit ist C_0 nur von M, also insbesondere von C_γ und γ abhängig. Nimmt man nun für diese Abschätzung das Supremum über alle S mit der Eigenschaft (3.38), so ergibt sich der erste Teil der Behauptung:

$$\phi(\rho, t) \leq C_0 \cdot t^u.$$

Wir beweisen nun die Umkehrung und nehmen an, dass ρ die ϕ-Bedingung (3.37) der Ordnung $u = \frac{1}{2-\gamma}$ für $\gamma \in (0;\beta)$ mit zugehöriger Konstanten $C_0 > 0$ erfüllt. Wir

wollen dann die wegen Lemma 3.6 äquivalente Bedingung (3.33)

$$\rho_{\mathbf{X}}\{x \in \mathbf{X}\colon |\eta(x)| \leq y\} \leq \bar{C}_\delta y^\delta, \quad \forall \delta \in (0;\alpha), \quad \forall y \in (0;1],$$

mit $\alpha := \frac{\beta}{1-\beta}$ zeigen. Sei $y \in (0;1]$, dann definieren wir

$$S := \{x \in \mathbf{X}\colon y/2 < |\eta(x)| \leq y\}$$

und $t := y^2\rho_S \in (0;1]$. Daraus ergibt sich

$$\int_S |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \leq \int_S y \,\mathrm{d}\rho_{\mathbf{X}} = y \cdot \rho_S = \sqrt{t\rho_S} \leq 3(t + \sqrt{t\rho_S}).$$

Das bedeutet, dass S eine zulässige Menge in der Definition von $\phi(\rho, t)$ aus (3.36) ist. Dann lässt sich aber die ϕ-Bedingung ausnutzen und wegen der Definition von S folgt

$$y/2 \cdot \rho_S = \int_S y/2 \,\mathrm{d}\rho_{\mathbf{X}} \leq \int_S |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \leq \phi(\rho, t) \leq C_0 t^u = C_0(y^2\rho_S)^u.$$

Das lässt sich zu

$$\rho_S^{1-u} \leq 2C_0 y^{2u-1}$$

und ferner mit

$$\frac{2u-1}{1-u} = \frac{\frac{2}{2-\gamma} - \frac{2-\gamma}{2-\gamma}}{\frac{2-\gamma}{2-\gamma} - \frac{1}{2-\gamma}} = \frac{\frac{\gamma}{2-\gamma}}{\frac{1-\gamma}{2-\gamma}} = \frac{\gamma}{1-\gamma},$$

zu

$$\rho_S \leq (2C_0)^{\frac{1}{1-u}} y^{\frac{2u-1}{1-u}} = (2C_0)^{\frac{1}{1-u}} y^{\frac{\gamma}{1-\gamma}} \tag{3.43}$$

umformen. Da $y \in (0;1]$ beliebig war, gilt die Ungleichung für die jeweilig zu y gehörigen Mengen S. Wir konstruieren nun die Menge $\{x \in \mathbf{X}\colon |\eta(x)| \leq y\}$. Für festes y sei

$$S_l := \left\{x \in \mathbf{X} \,\middle|\, \frac{y}{2^{l+1}} < |\eta(x)| \leq \frac{y}{2^l}\right\}.$$

Dann sind die S_l paarweise disjunkt und es ist

$$S_{(y)} := \{x \in \mathbf{X}\colon |\eta(x)| \leq y\} = \underbrace{\{x \in \mathbf{X}\colon |\eta(x)| = 0\}}_{=:\mathbf{X}_{\eta=0}} \cup \bigcup_{l \in \mathbb{N}_0} S_l.$$

Setzt man $C := (2C_0)^{\frac{1}{1-u}}$ und $\gamma := \frac{\delta}{1+\delta}$ bzw. $\delta = \frac{\gamma}{1-\gamma}$, so folgt

$$\begin{aligned}
\rho_{S_{(y)}} = \int\limits_{S_{(y)}} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} &= \int\limits_{\mathbf{X}_{\eta=0}} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} + \sum_{l=0}^{\infty} \int\limits_{S_l} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \\
&\leq 0 + \sum_{l=0}^{\infty} \int\limits_{S_l} \frac{y}{2^l} \,\mathrm{d}\rho_{\mathbf{X}} \\
&= 0 + \sum_{l=0}^{\infty} \frac{y}{2^l} \cdot \rho_{S_l} \\
&\leq \sum_{l=0}^{\infty} \rho_{S_l} \overset{(3.43)}{\leq} \sum_{l=0}^{\infty} C \cdot \left(\frac{y}{2^l}\right)^{\delta} \\
&\leq C \cdot y^{\delta} \cdot \sum_{l=0}^{\infty} \left(\frac{1}{2^{\delta}}\right)^{l} \\
&= C \cdot y^{\delta} \cdot \frac{1}{1-2^{-\delta}}.
\end{aligned}$$

Mit $\bar{C}_\delta := C \cdot \frac{1}{1-2^{-\delta}}$ für $\delta \in (0;\alpha)$ ist dann (3.33) erfüllt. Somit wurde auch der zweite Teil der Behauptung bewiesen. □

3.3 Obere Schranke für den Näherungsfehler

Der Näherungsfehler hängt vom Wahrscheinlichkeitsmaß ρ und der Reichhaltigkeit der Familie $\mathcal{S}$ ab. Typischerweise benutzt man Folgen $(\mathcal{S}_m)_{m=1}^{\infty}$ von ineinander geschachtelten Familien von Mengen: $\mathcal{S}_m \subseteq \mathcal{S}_{m+1}$ für $m \in \mathbb{N}$. Der Wert m und die Familie $\mathcal{S}_m$, die für die konkreten Beobachtungsdaten benutzt werden, hängen von der Anzahl der Beobachtungen und eventuell bekannten Eigenschaften wie der Glattheit der Regressionsfunktion η und Tsybakov-Bedingungen vom Wahrscheinlichkeitsmaß ρ ab. Dazu benutzt man eine Form der Modellauswahl. Um die Leistung, der auf diese Art erzeugten Algorithmen, zu analysieren, hätte man gerne Bedingungen an ρ, die das Verhalten des Näherungsfehlers für m gegen unendlich steuern.

Definition 3.10. Wir definieren den Fehler

$$a_m(\rho) := a(\Omega^*, \mathcal{S}_m), \qquad \text{für } m \in \mathbb{N}.$$

Er ist wegen der Ineinanderschachtelung der Familien $\mathcal{S}_m$, als Funktion von m betrachtet, monoton fallend.

Definition 3.11. Ferner definieren wir die Approximationsklasse

$$\mathcal{A}^s := \mathcal{A}^s((\mathcal{S}_m)_{m=1}^\infty)$$

als die Menge aller Wahrscheinlichkeitsmaße ρ für die

$$|\rho|_{\mathcal{A}^s} := \sup_{m\geq 1} m^s a_m(\rho), \qquad s > 0, \tag{3.44}$$

endlich ist.

Unser Ziel ist es nun zu verstehen, welche Eigenschaften von ρ eine Zugehörigkeit zur Approximationsklasse $\mathcal{A}^s$ garantieren. Dazu geben wir hinreichende Bedingungen an, die mit der Glattheit der Regressionsfunktion η und Tsybakov-Bedingungen verknüpft sind.

Sei $(\mathcal{S}_m)_{m=1}^\infty$ eine monotone Folge von Mengensystemen. Wir fixieren nun ein ρ. Dadurch sind die Regressionsfunktion η und die Bayes'sche Menge

$$\Omega^* = \{x \in \mathbf{X}\colon \eta(x) \geq 0\}$$

bestimmt. Weiter definieren wir die Niveaumenge

$$\Omega(t) := \{x \in \mathbf{X}\colon \eta(x) \geq t\}.$$

Daraus folgt für $t \geq t'$ sofort $\Omega(t) \subseteq \Omega(t')$ und dann mit $t > 0$

$$\begin{aligned}\{x \in \mathbf{X}\colon |\eta(x)| < t\} &\subseteq \{x \in \mathbf{X}\colon -t \leq \eta(x) \wedge \eta(x) < t\}\\ &= \Omega(-t)\backslash\Omega(t)\\ &\subseteq \{x \in \mathbf{X}\colon |\eta(x)| \leq t\}.\end{aligned}$$

Außerdem definieren wir für $m \in \mathbb{N}$

$$t_m := t_m(\rho, \mathcal{S}_m) := \inf\{t > 0 \mid \exists S \in \mathcal{S}_m\colon \Omega(t) \subseteq S \subseteq \Omega(-t)\}.$$

Der Einfachheit halber nehmen wir an, dass immer eine Menge $S_m^* \in \mathcal{S}_m$ mit $\Omega(t_m) \subseteq S_m^* \subseteq \Omega(-t_m)$ existiert. Andernfalls ersetzen wir t_m durch $t_m + \varepsilon$ mit $\varepsilon > 0$ beliebig klein und erhalten dafür Mengen S_m^*. Die nachfolgenden Ergebnisse erhält man trotzdem. Da auch $\Omega(t_m) \subseteq \Omega^* \subseteq \Omega(-t_m)$ gilt, folgt

$$\Omega^* \Delta S_m^* \subseteq \Omega(-t_m)\backslash\Omega(t_m).$$

Erfüllt ρ die Tsybakov-Bedingung (3.33) für ein $\alpha \in (0;\infty)$, so gilt für alle $\delta \in (0;\alpha)$ die Ungleichung

$$\begin{aligned} a_m(\rho) &= R(\Omega_{\mathcal{S}_m}) - R(\Omega^*) \le R(S_m^*) - R(\Omega^*) \\ &\overset{(2.2)}{=} \int\limits_{\Omega^*} \eta \,\mathrm{d}\rho_{\mathbf{X}} - \int\limits_{S_m^*} \eta \,\mathrm{d}\rho_{\mathbf{X}} \\ &= \int\limits_{\Omega^* \Delta S_m^*} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \le \int\limits_{\Omega(-t_m)\setminus\Omega(t_m)} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \\ &\le \int\limits_{\{x\in\mathbf{X}\colon\, |\eta(x)|\le t_m\}} |\eta| \,\mathrm{d}\rho_{\mathbf{X}} \le \int\limits_{\{x\in\mathbf{X}\colon\, |\eta(x)|\le t_m\}} t_m \,\mathrm{d}\rho_{\mathbf{X}} \\ &\overset{(3.33)}{\le} \bar{C}_\delta t_m \cdot t_m^\delta = \bar{C}_\delta t_m^{\delta+1}. \end{aligned} \tag{3.45}$$

Somit ist $t_m^{\delta+1} \le C m^{-s}$ für ein $\delta \in (0;\alpha)$ in Verbindung mit der Tsybakov-Bedingung (3.33) für $\alpha \in (0;\infty)$ und $0 < C < \infty$ hinreichend für die Zugehörigkeit von ρ zur Approximationsklasse $\mathcal{A}^s$.

Eine Anwendung dieser Bedingungen soll im Folgenden gezeigt werden. Sei $\mathbf{X} = [0;1)^d$ und $\mathcal{D}$ das System der dyadischen Würfel Q, die in $\mathbf{X}$ enthalten sind. Das sind Würfel $Q \subseteq \mathbf{X}$ der Form $Q = 2^{-j}(k+[0;1)^d)$ mit $k \in \mathbb{Z}^d$ und $j \in \mathbb{N}_0$. Ferner sei $\mathcal{D}_j$ für $j \in \mathbb{N}_0$ das Mengensystem der dyadischen Würfel mit Seitenlänge 2^{-j} und $\mathcal{S}_{2^{dj}}$ das Mengensystem aller Mengen der Form $S_\Lambda = \bigcup\limits_{Q\in\Lambda} Q$ mit $\Lambda \subseteq \mathcal{D}_j$. Man beachte, dass $|\mathcal{D}_j| = 2^{dj}$ und $|\mathcal{S}_{2^{dj}}| = 2^{2^{dj}}$ gilt. Wir erzeugen eine Familie von Klassen $(S_m)_{m\in\mathbb{N}}$ durch $\mathcal{S}_m := \mathcal{S}_{2^{dj}}$, wenn $2^{dj} \le m < 2^{d(j+1)}$. Für unser Wahrscheinlichkeitsmaß ρ nehmen wir nun an, dass es den nachfolgenden zwei Bedingungen genüge:

- Die Regressionsfunktion η liegt im Lipschitzraum (oder Hölderraum) $Lip\, b$ für ein $b \in (0;1]$. Das heißt, dass Folgendes gilt

  $$|\eta|_{Lip\, b} := \sup\left\{|\eta(x) - \eta(\tilde{x})| \cdot ||x-\tilde{x}||_2^{-b} \colon x, \tilde{x} \in \mathbf{X}\right\} < \infty,$$

  wobei $||\cdot||_2$ die euklidische Norm bezeichnet.

- Für ρ gilt die Bedingung (3.33) für $\alpha \in (0,\infty)$.

Wir behaupten, dass

$$a_{2^{dj}}(\rho) \le \bar{C}_\delta \cdot (M 2^{-jb})^{\delta+1}, \qquad j \ge 0,$$

mit $M := 2^{-b} d^{b/2} |\eta|_{Lip\, b} < \infty$ erfüllt ist.

Sei $Q \in \mathcal{D}_j$ und ξ_Q das Zentrum von Q. Der maximale euklidische Abstand vom Zentrum eines Würfels zu einem Punkt des Würfels wird in einer Ecke erreicht. Damit gilt für jedes $x \in Q$:

$$\begin{aligned} |\eta(x) - \eta(\xi_Q)| &= |\eta(x) - \eta(\xi_Q)| \cdot \frac{||x - \xi_Q||_2^b}{||x - \xi_Q||_2^b} \\ &\leq ||x - \xi_Q||_2^b \cdot |\eta|_{Lip\, b} \\ &\leq \left(\sqrt{\sum_{l=1}^{d} \left(2^{-j-1}\right)^2} \right)^b \cdot |\eta|_{Lip\, b} \\ &= \left(d^{1/2} \cdot 2^{-j-1}\right)^b \cdot |\eta|_{Lip\, b} \\ &= M \cdot 2^{-jb}. \end{aligned}$$

Wir definieren S_j als die Vereinigung aller $Q \in \mathcal{D}_j$, die $\eta(\xi_Q) \geq 0$ genügen. Somit ist $S_j \in \mathcal{S}_{2^{dj}}$. Sei $t := M \cdot 2^{-jb}$, dann ist

$$\Omega(t) \subseteq S_j \subseteq \Omega(-t), \qquad j \in \mathbb{N}_0,$$

denn ist $x \in \Omega(t)$, so folgt $\eta(x) \geq t$. Zu diesem x gibt es ein $Q \in \mathcal{D}_j$ mit $x \in Q$. Da $\eta(\xi_Q) < 0$ im Widerspruch zu der eben gezeigten Ungleichung für unser x stehen würde, folgt dann $\eta(\xi_Q) \geq 0$. Also ist $Q \subseteq S_j$ und damit $x \in S_j$. Das zeigt $\Omega(t) \subseteq S_j$. Ähnlich folgt der zweite Teil. Ist $x \in S_j$, dann existiert ein $Q \in \mathcal{D}_j$ mit $x \in Q \subseteq S_j$ und $\eta(\xi_Q) \geq 0$. Deswegen folgt

$$-t \leq \eta(x) - \eta(\xi_Q) \leq \eta(x).$$

Ferner erhält man daraus $x \in \Omega(-t)$ und so auch $S_j \subseteq \Omega(-t)$. Unsere Behauptung folgt nun analog zu (3.45), wobei wir m durch 2^{dj}, $\mathcal{S}_m$ durch $\mathcal{S}_{2^{dj}}$, t_m durch t und S_m^* durch S_j ersetzen.

Das Beispiel zeigt, dass die Kombination der Tsybakovbedingung (3.33) der Ordnung β mit Hölderglattheit der Ordnung b für die Regressionsfunktion η dazu führt, dass ρ zur Approximationsklasse $\mathcal{A}^s = \mathcal{A}^s((S_{2^{dj}}))$ mit $s := \frac{b(\delta+1)}{d}$ für ein $\delta \in (0; \alpha)$ gehört. Denn mit $m = 2^{dj}$ ist

$$m^s a_m(\rho) = (2^{dj})^{b(\delta+1)/d} \cdot a_{2^{dj}}(\rho) \leq (2^{jb})^{(\delta+1)} \cdot \bar{C}_\delta \cdot (M 2^{-jb})^{\delta+1} = \bar{C}_\delta \cdot M^{\delta+1} < \infty$$

und somit $|\rho|_{\mathcal{A}^s}$ endlich.

4 Modellauswahl

Die hier vorgestellte Modellauswahl stammt aus [Bin+14a, S. 2153 ff.]. Sie benutzt die Schranken aus Kapitel 3, um eine vorteilhafte Abschätzung für das zusätzliche Risiko unseres ausgewählten Klassifikators zu erhalten.

Die Idee hinter unserer Modellauswahl ist die Unterteilung der Beobachtungsdaten in zwei Gruppen. Die erste Gruppe wird genutzt, um eine gewisse Anzahl an Klassifikatoren zu bilden, aus der im zweiten Schritt der empirisch Beste ausgewählt wird. Die empirische Auswahl erfolgte dabei durch die zweite Gruppe von Beobachtungsdaten, wobei meist das empirische Risiko (2.6) als Kriterium genutzt wird.

Sei $(\mathcal{S}_m)_{m=1}^{\infty}$ eine Folge von Familien von Mengen, die benutzt wird, um einen binären Klassifikationsalgorithmus zu entwickeln. Wir nehmen an, dass eine Konstante $C_0 > 0$ mit

$$VC(\mathcal{S}_m) \leq C_0 m, \qquad m \in \mathbb{N},$$

existiert und bezeichnen $\bar{\Omega}_m$ als den Klassifikator in $\mathcal{S}_m$, der das empirische Risiko minimiert und mit $\hat{\eta}_S = \bar{\eta}_S$ durch (2.4) gewählt wird. Wir haben in Satz 3.1 gezeigt, dass solch ein Schätzer eine Schranke (3.2)

$$|\eta_S - \eta_{\Omega_S} - (\bar{\eta}_S - \bar{\eta}_{\Omega_S})| \leq e_n(S)$$

mit $S \in \mathcal{S}_m$ und Wahrscheinlichkeit mindestens $1 - C_1 n^{-r}$ erfüllt. Dabei ist

$$e_n(S) = \sqrt{\rho_{S\Delta\Omega_S} \cdot \varepsilon_n} + \varepsilon_n, \qquad \varepsilon_n = C_2 \cdot \frac{m \ln n}{n}$$

und C_2 ist abhängig von r, B, C_0, sowie C_1 eine absolute Konstante. Ist $\rho \in \mathcal{A}^s((\mathcal{S}_m))$ für ein $s > 0$, so gilt nach Folgerung 3.5 für jedes $m \in \mathbb{N}$ mit Wahrscheinlichkeit von mindestens $1 - C_1 n^{-r}$

$$\begin{aligned} R(\bar{\Omega}_m) - R(\Omega^*) &\leq \omega(\rho, e_n) + 2a(\Omega^*, \mathcal{S}_m) \\ &\leq \omega(\rho, e_n) + 2|\rho|_{\mathcal{A}^s} m^{-s}. \end{aligned} \tag{4.1}$$

Erfüllt ρ zusätzlich auch die Tsybakov-Bedingung der Ordnung β, dann erhält man durch Ausnutzung von Satz 3.9 und der Tatsache, dass

$$\omega(\rho, e_n) \leq \phi(\rho, \varepsilon_n) \leq C_3 \varepsilon_n^{\frac{1}{2-\gamma}}$$

für jedes $\gamma \in (0;\beta)$ gilt, mit jeweiliger Wahrscheinlichkeit das Folgende aus (4.1):

$$R(\bar{\Omega}_m) - R(\Omega^*) \leq C_4 \cdot \left(\frac{m \ln n}{n}\right)^{\frac{1}{2-\gamma}} + 2|\rho|_{\mathcal{A}^s} m^{-s}, \tag{4.2}$$

wobei C_4 abhängig von C_γ und γ ist. Man beachte die durch die verschiedenen Ereignisse wichtigen folgenden Beziehungen, wobei A_k Ereignisse und $P_k \in [0;1]$ zugehörige Konstanten sind:

$$\mathbf{P}\left[A_k\right] \geq 1 - P_k \Rightarrow 1 - \mathbf{P}\left[A_k\right] = \mathbf{P}\left[A_k^C\right] \leq P_k$$

$$\Rightarrow \mathbf{P}\left[\bigcup_{k=1}^{l} A_k^C\right] \leq \sum_{k=1}^{l} \mathbf{P}\left[A_k^C\right] \leq \sum_{k=1}^{l} P_k$$

$$\Rightarrow \mathbf{P}\left[\bigcap_{k=1}^{l} A_k\right] = 1 - \mathbf{P}\left[\left(\bigcap_{k=1}^{l} A_k\right)^C\right] = 1 - \mathbf{P}\left[\bigcup_{k=1}^{l} A_k^C\right] \geq 1 - \sum_{k=1}^{l} P_k. \tag{4.3}$$

Wir unterteilen die Ziehung $\mathbf{z}$ in zwei unabhängige Mengen $\mathbf{z}'$ und $\mathbf{z}''$ von gleicher Größe $\bar{n}$. Für jedes m mit $1 \leq m \leq \bar{n}$ sei $\bar{\Omega}_m$ durch (2.4) definiert, wobei $\mathcal{S}$ durch $\mathcal{S}_m$ und $\mathbf{z}$ durch $\mathbf{z}'$ ersetzt wurde. Für jedes m erfüllt $\bar{\Omega}_m$ (4.2) mit $\bar{n}$ anstatt n und Wahrscheinlichkeit von mindestens $1 - C_1\bar{n}^{-r}$ für das jeweilige m. Dann erhalten wir mit jeweiliger Wahrscheinlichkeit

$$R(\bar{\Omega}_m) - R(\Omega^*) \leq C_5 \cdot \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} + m^{-s}\right), \qquad m = 1, \ldots, \bar{n}.$$

Wir setzen nun $\bar{\mathcal{S}} := \left\{\bar{\Omega}_1, \ldots, \bar{\Omega}_{\bar{n}}\right\}$ und bezeichnen mit $\bar{\Omega}_{m^*}$ die Menge, welche aus $\bar{\mathcal{S}}$ durch (2.4) unter Benutzung von $\mathbf{z}''$ ausgewählt wird. Aus den $\bar{n}$ vorhergehenden Ungleichungen folgt

$$a(\Omega^*, \bar{\mathcal{S}}) = \min_{1 \leq m \leq \bar{n}} \left(R(\bar{\Omega}_m) - R(\Omega^*)\right) \leq C_5 \min_{1 \leq m \leq \bar{n}} \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} + m^{-s}\right),$$

wegen (4.3) mit Wahrscheinlichkeit von mindestens

$$1 - \sum_{m=1}^{\bar{n}} C_1 \bar{n}^{-r} = 1 - C_1 n^{-r+1}. \tag{4.4}$$

Es sei

$$\bar{\varepsilon}_{\bar{n}} := \frac{14(r \ln \bar{n} + \ln|\bar{S}|)}{3 \cdot \bar{n}}, \qquad \bar{e}_{\bar{n}}(S) := \sqrt{\rho_{S\Delta\Omega_{\bar{S}}} \cdot \bar{\varepsilon}_{\bar{n}}} + \bar{\varepsilon}_{\bar{n}}.$$

Da $|\bar{\mathcal{S}}| = \bar{n} = n/2$ ist, ergibt sich $\bar{\varepsilon}_{\bar{n}} \leq C_6 \cdot \frac{\ln \bar{n}}{\bar{n}}$ in (3.19) bei Benutzung von $\bar{\mathcal{S}}$ und (3.20) mit Wahrscheinlichkeit von mindestens $1 - 2\bar{n}^{-r}$. Unter Ausnutzung von Folgerung 3.5 und Satz 3.9, sowie der Tatsache, dass

$$\omega(\rho, \bar{e}_{\bar{n}}) \leq \phi(\rho, \bar{\varepsilon}_{\bar{n}}) \leq C_7 \cdot \bar{\varepsilon}_{\bar{n}}^{\frac{1}{2-\gamma}}$$

für jedes $\gamma \in (0; \beta)$ gilt, erhalten wir daraus

$$\begin{aligned} R(\bar{\Omega}_{m^*}) - R(\Omega^*) &\leq 2a(\Omega^*, \bar{\mathcal{S}}) + C_8 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} \\ &\leq C_5 \min_{1 \leq m \leq \bar{n}} \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} + m^{-s} \right) + C_8 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} \end{aligned} \tag{4.5}$$

wegen (4.4) und der Unabhängigkeit von $\mathbf{z}'$ und $\mathbf{z}''$ mit Wahrscheinlichkeit von mindestens

$$\begin{aligned} (1 - C_1 n^{-r+1})(1 - 2n^{-r}) &= 1 - 2n^{-r} - C_1 n^{-r+1} + 2C_1 n^{-2r+1} \\ &\geq 1 - 2n^{-r} - C_1 n^{-r+1} \\ &\geq 1 - 2n^{-r+1} - C_1 n^{-r+1} \\ &= 1 - (2 + C_1) n^{-r+1} \\ &= 1 - C n^{-r+1}. \end{aligned}$$

Dabei ist $C := 2 + C_1$. Für $u := \frac{1}{2-\gamma}$ ist

$$\begin{aligned} m_0 &:= \left(\frac{s}{u}\right)^{\frac{1}{s+u}} \cdot \left(\frac{\bar{n}}{\ln \bar{n}}\right)^{\frac{u}{u+s}} \\ &= (s(2-\gamma))^{\frac{2-\gamma}{s(2-\gamma)+1}} \cdot \left(\frac{\bar{n}}{\ln \bar{n}}\right)^{\frac{1}{s(2-\gamma)+1}} \end{aligned}$$

und wegen Lemma A.21

$$\begin{aligned} &\min_{1 \leq m \leq \bar{n}} \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{u} + m^{-s} \right) \\ &= \min_{m \in \{\lceil m_0 \rceil; \lfloor m_0 \rfloor\}} \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{u} + m^{-s} \right) \\ &\sim \left(\frac{m_0 \ln \bar{n}}{\bar{n}}\right)^{u} + m_0^{-s} \\ &= \left(\frac{s}{u}\right)^{\frac{u}{s+u}} \left(\frac{\bar{n}}{\ln \bar{n}}\right)^{\frac{u^2}{s+u}} \cdot \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{u} + \left(\frac{s}{u}\right)^{-\frac{s}{s+u}} \cdot \left(\frac{\bar{n}}{\ln \bar{n}}\right)^{-\frac{su}{u+s}} \\ &= \left(\frac{s}{u}\right)^{\frac{u}{s+u}} \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{-\frac{u^2}{s+u}} \cdot \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{us+u^2}{s+u}} + \left(\frac{s}{u}\right)^{-\frac{s}{s+u}} \cdot \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{su}{u+s}} \end{aligned} \tag{4.6}$$

$$= \left[\left(\frac{s}{u}\right)^{\frac{u}{s+u}} + \left(\frac{s}{u}\right)^{-\frac{s}{s+u}} \right] \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{su}{u+s}}$$
$$= C_9 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{s}{1+s(2-\gamma)}}.$$

Aus (4.6) und (4.5) folgt dann mit Wahrscheinlichkeit von mindestens $1 - Cn^{-r+1}$

$$\begin{aligned} R(\bar{\Omega}_{m^*}) - R(\Omega^*) &\leq C_5 \min_{1 \leq m \leq \bar{n}} \left(\left(\frac{m \ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} + m^{-s} \right) + C_8 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} \\ &\sim C_5 C_9 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{s}{(2-\gamma)s+1}} + C_8 \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{1}{2-\gamma}} \\ &\leq \bar{C} \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{s}{(2-\gamma)s+1}}. \end{aligned}$$

Die letzte Ungleichung gilt wegen

$$\frac{s}{(2-\gamma)s+1} = \frac{1}{(2-\gamma)+\frac{1}{s}} \leq \frac{1}{2-\gamma}$$

und

$$0 \leq \frac{\ln \bar{n}}{\bar{n}} \leq 1, \qquad n \geq 2,$$

sowie der Monotonie der Funktion $f(x) := y^x$ mit $y \in [0;1]$, welche für $x > 0$ monoton fällt.

Insgesamt liefern diese Überlegungen den folgenden

Satz 4.1. *Sei $(\mathcal{S}_m)_{m=1}^{\infty}$ eine Folge von Familien von $\rho_{\mathbf{X}}$-messbaren Teilmengen von $\mathbf{X}$, die für eine Konstante $C_0 > 0$*

$$VC(\mathcal{S}_m) \leq C_0 m$$

für alle $m \in \mathbb{N}$ erfüllt. Ist außerdem $\rho \in \mathcal{A}^s((\mathcal{S}_m))$ für ein $s > 0$ und erfüllt ρ die Tsybakov-Bedingung der Ordnung β, so existieren zu jedem $r > 0$ Konstanten $C, \bar{C} > 0$, sodass

$$R(\bar{\Omega}_{m*}) - R(\Omega^*) \leq \bar{C} \left(\frac{\ln \bar{n}}{\bar{n}}\right)^{\frac{s}{(2-\gamma)s+1}} \tag{4.7}$$

asymptotisch mit Wahrscheinlichkeit von mindestens $1 - Cn^{-r+1}$ für $\gamma \in (0;\beta)$ gilt.

In der Praxis sind die Parameter s und γ meist unbekannt. Unser hier erhaltenes Resultat besagt nun, dass wir durch unsere einfache Modellauswahl auch ohne Kenntnis dieser Parameter einen Klassifizierer finden, der für hinreichend großes n mit großer Wahrscheinlichkeit vom optimalen Klassifikator Ω^* bezüglich dem Risiko kaum zu unterscheiden ist.

5 Anwendung auf Partitionierungen mittels dyadischen Bäumen

In diesem Kapitel werden die theoretischen Ergebnisse der vorherigen Kapitel auf ein konkretes Klassifizierungsverfahren angewendet. Dieses Verfahren werden wir im Folgenden „Classification using dyadic tree partitioning“ oder auch CUDTP nennen. Die Idee dazu stammt aus [Bin+14a, S. 2156 f.].

Versucht man die Menge Ω^*, die den Bayes'schen Klassifikator erzeugt, nachzubilden, so ist eine der natürlichsten Vorgehensweisen mittels adaptiver Partitionierung. Diese Art von Partitionierung besitzt die Flexibilität den Bereich des Randes von Ω^* fein aufzulösen und alle anderen Bereiche nur grob, vgl. [Bin+14a, S. 2156]. Zur Vereinfachung der Darstellung betrachten wir nur dyadische Partitionierungen auf dem speziellen Definitionsbereich $\mathbf{X} = [0;1)^d$.

Zur Beschreibung des Verfahrens benutzen wir die Begrifflichkeiten aus [Bin+14a, S. 2156] und für den Begriff des Baumes verweisen wir auf [Ern00, S.585]: Sei $\mathcal{D}$ das Mengensystem der dyadischen Würfel Q, die in X enthalten sind. Das sind Mengen $Q \subseteq \mathbf{X}$ der Form

$$Q = 2^{-j}(k + [0;1)^d) \qquad \text{mit } k \in \mathbb{Z}^d, j \in \mathbb{Z}.$$

Ferner seien $\mathcal{D}_j$ mit $j \in \mathbb{N}_0$ Mengensysteme der dyadischen Würfel mit Seitenlänge 2^{-j}. Jedes $Q \in \mathcal{D}_j$ hat 2^d Nachfolger, die Teilmengen von Q aus $\mathcal{D}_{j+1}$ sind. Somit bilden diese Würfel einen Baum mit Wurzel $\mathbf{X}$. Ein endlicher Teilbaum $\mathcal{T}$ von $\mathcal{D}$ ist eine endliche Menge von Würfeln mit der Eigenschaft, dass die Wurzel $\mathbf{X}$ in $\mathcal{T}$ liegt und falls Q ein Element von $\mathcal{T}$ ist, so ist es auch der Vorgänger von Q. Wir sagen ein Baum ist vollständig, falls gilt, dass immer wenn Q in $\mathcal{T}$ liegt, dann liegen auch alle Geschwister von Q in $\mathcal{T}$. Geschwister von Q sind dabei die Würfel, die den gleichen Vorgänger wie Q besitzen. Die Menge der Blätter eines Baumes $\mathcal{T}$ besteht aus allen Würfeln $Q \in \mathcal{T}$, die keinen Nachfolger in $\mathcal{T}$ haben und wird mit $\mathcal{L}(\mathcal{T})$ bezeichnet. Ist ein endlicher Baum vollständig, so bildet die Menge aller Blätter eine Partition von $\mathbf{X}$.

Jeder endliche und vollständige Baum ist das Ergebnis einer endlichen Anzahl von aufeinanderfolgenden Würfelverfeinerungen. Wir bezeichnen mit $\mathfrak{T}_m$ das Mengensystem aller durch m Verfeinerungen entstandenen vollständigen Bäume. Jeder Baum $\mathcal{T} \in \mathfrak{T}_m$ besitzt $(2^d - 1)m + 1$ Blätter, denn bei jeder Verfeinerung verschwindet ein Blatt, aus dem 2^d neue Blätter entstehen. Durch Zuweisung einer eindeutigen

Folge von Bits für jeden Baum lässt sich die Anzahl der Bäume $\mathcal{T} \in \mathfrak{T}_m$ abschätzen. Sei $\mathcal{T} \in \mathfrak{T}_m$. Wir ordnen die Nachfolger von $\mathbf{X}$ lexikografisch und weisen für jeden solchen Nachfolger in $\mathcal{T}$ eine Eins zu, falls dieser verfeinert wird, ansonsten eine Null. Analog gehen wir für die Nachfolger der Nachfolger aus $\mathcal{T}$ vor. Diese sind durch die bereits zugewiesenen Bits definiert. Das Verfahren wird fortgesetzt, bis die Blätter von $\mathcal{T}$ erreicht werden. Somit hat man eine Folge von Bits erhalten, die den Baum $\mathcal{T}$ eindeutig bestimmt. Da $\mathcal{T}$ aus genau $2^d m+1$ Würfeln besteht, hat die zugehörige Bitfolge eine Länge von $2^d m$ und eine Eins genau an $m-1$ Positionen. Abbildung 2 illustriert das Verfahren für eine Baum mit $d = 2$ und $m = 4$. Dabei gehört 1010 zu $Q_{1,1}$ bis $Q_{1,4}$, die darauf folgenden vier Bits 0000 zu $Q_{2,1}$ bis $Q_{2,4}$, 0100 zu $Q_{2,5}$ bis $Q_{2,8}$ und die letzten vier Bits 0000 zu $Q_{3,1}$ bis $Q_{3,4}$.

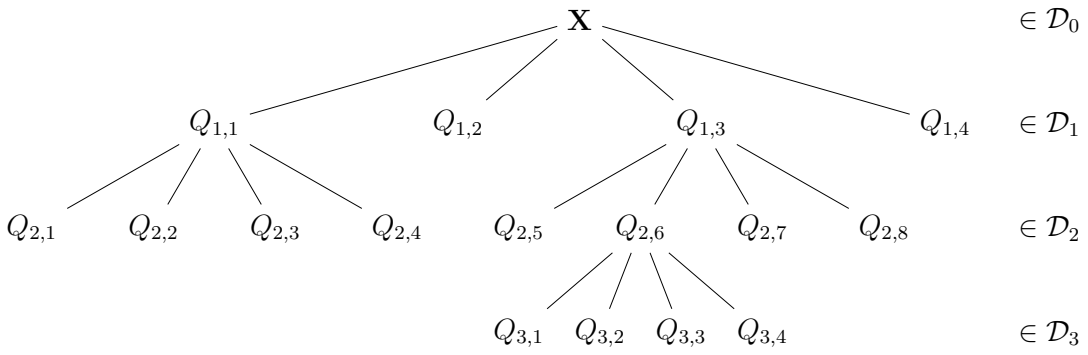


Abbildung 2: Baum, der bei $d = 2$ und $m = 4$ zur Bitfolge 1010 0000 0100 0000 führt.

Die Anzahl verschiedener Bitfolgen mit Länge $2^d m$ und $m-1$ Einsen, ergibt sich aus der Anzahl an Kombinationen ohne Wiederholung von $m-1$ aus $2^d m$ Elementen. Da nicht alle solche Bitfolgen einen Baum bilden, lässt sich die Anzahl der Bäume durch die Anzahl dieser Bitfolgen wie folgt abschätzen:

$$|\mathfrak{T}_m| \leq \binom{2^d m}{m-1} = \frac{1}{(m-1)!} \cdot \prod_{i=1}^{m-1} (2^d m - i + 1) \leq \frac{(2^d m)^{m-1}}{(m-1)!} \leq e^m \cdot 2^{d(m-1)}. \quad (5.1)$$

Das letzte Ungleichungszeichen folgt aus der Reihenentwicklung der Exponentialfunktion, denn für m als Argument sind alle Summanden nichtnegativ und auf der linken Seite steht der Summand für $k = m-1$ multipliziert mit einer Konstanten:

$$\frac{m^{m-1}}{(m-1)!} \leq \sum_{k=0}^{\infty} \frac{m^k}{k!} = \exp(m) = e^m.$$

Wir definieren nun für jede Wahl eines Baumes $\mathcal{T} \in \mathfrak{T}_m$ und $\Lambda \subseteq \mathcal{L}(\mathcal{T})$ eine Menge $S := S_\Lambda := \bigcup_{Q \in \Lambda} Q$, die später einen möglichen Kandidat für einen Klassifi-

zierer darstellt. Weiterhin bezeichnen wir für die Modellauswahl das System aller solcher Mengen S für $\mathcal{T} \in \mathfrak{T}_m$ und $\Lambda \subseteq \mathcal{L}(\mathcal{T})$ mit $\mathcal{S}_m$. Hat man einen Baum $\mathcal{T}$ ausgewählt, so ergibt sich für die Anzahl der verschiedenen Teilmengen Λ von $\mathcal{L}(\mathcal{T})$:

$$|\mathfrak{P}(\mathcal{L}(\mathcal{T}))| = 2^{|\mathcal{L}(\mathcal{T})|} \leq 2^{2^d m}, \tag{5.2}$$

wobei $\mathfrak{P}$ die Potenzmenge bezeichne. Kombiniert man nun die beiden Abschätzungen (5.1) und (5.2), so ergibt sich:

$$\begin{aligned} |\mathcal{S}_m| &= \left| \bigcup_{\mathcal{T} \in \mathfrak{T}_m} \mathfrak{P}(\mathcal{L}(\mathcal{T})) \right| \\ &\leq |\mathfrak{T}_m| \cdot |\mathfrak{P}(\mathcal{L}(\mathcal{T}))| \leq |\mathfrak{T}_m| \cdot 2^{2^d m} \\ &\leq e^m \cdot 2^{d(m-1)} \cdot 2^{2^d m} \leq e^m \cdot 2^{(d+2^d)m} \\ &= a^m. \end{aligned} \tag{5.3}$$

Dabei ist $a := e \cdot 2^{d+2^d}$.

Benutzt man den Mengenschätzer und die Modellwahl über $(\mathcal{S}_m)_{m \in \mathbb{N}}$, wie im vorherigen Kapitel für die Ziehung $\mathbf{z}$ beschrieben, so erhält man eine Menge $\bar{\Omega}(\mathbf{z})$. Die Leistung des zugehörigen Klassifikators lässt sich dann wie folgt beurteilen.

Satz 5.1. *Für jedes $r > 0$ existiert eine Konstante $\bar{C} > 0$, so dass das Folgende gilt: Ist $\rho \in \mathcal{A}^s$ mit $s > 0$ und ρ erfüllt die Tsybakov-Bedingung (3.32) der Ordnung β, dann erhält man für jedes $\gamma \in (0; \beta)$ mit Wahrscheinlichkeit mindestens $1 - C\bar{n}^{-r+1}$ die asymptotisch erfüllte Ungleichung*

$$R(\bar{\Omega}(\mathbf{z})) - R(\Omega^*) \leq \bar{C} \left(\frac{\ln \bar{n}}{\bar{n}} \right)^{\frac{s}{(2-\gamma)s+1}}, \tag{5.4}$$

wobei $\bar{C}$ von $d, r, A, |\rho|_{\mathcal{A}^s}$ und den Konstanten aus (3.32) abhängt. $C > 0$ ist dabei eine Konstante unabhängig von r.

Beweis. Da $|\mathcal{S}_m| \leq a^m < \infty$ gilt, erhält man mit Lemma 2.10

$$VC(\mathcal{S}_m) \leq \log_2(|\mathcal{S}_m|) \leq m \log_2(a) =: C_0 m,$$

wobei C_0 nur abhängig von d nach (5.3) ist. Somit wird $R(\bar{\Omega}(\mathbf{z})) - R(\Omega^*)$ durch die rechte Seite von (4.7) aus Satz 4.1 beschränkt. □

6 Implementierung von CUDTP und ODT

Dieser Abschnitt wird genutzt, um die Ideen der implementierten Algorithmen zu erläutern und auf eventuelle Veränderungen einzugehen.

6.1 Classification Using Dyadic Tree Partitioning (CUDTP)

In Kapitel 5 wurde gezeigt, dass unser Hauptresultat (4.7) zur Abschätzung des zusätzlichen Risikos auch für Partitionen mittels dyadischen Entscheidungsbäumen anwendbar ist, siehe Satz 5.1. Die Implementierungsidee von „Classification using dyadic tree partitioning“ stammt aus [Bin+14a, S. 2160 ff.] und behandelt das Problem, die Mengen $\bar{\Omega}_m \in \mathcal{S}_m$ von (2.4) möglichst effizient zu finden. Diese werden dann bei der Modellauswahl zur Bildung des Klassifizerers $\bar{\Omega}_{m^*}$ genutzt. Er erfüllt dann die wichtige Abschätzung für das zusätzliche Risiko (4.7) bzw. (5.4).

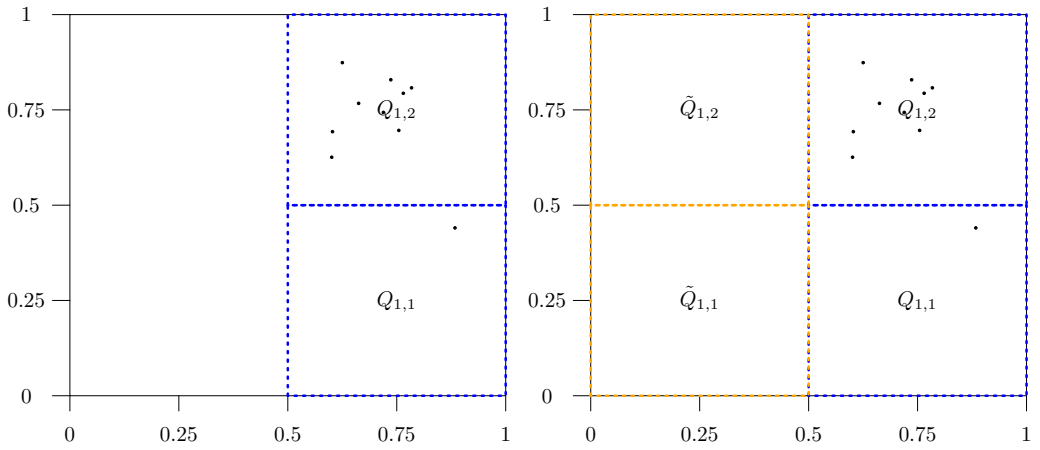
Bei gegebenen Beobachtungsdaten $\mathbf{z}$ betrachten wir das System der dyadischen Würfel aus $\mathcal{D}_0 \cup \cdots \cup \mathcal{D}_{\bar{n}}$ mit $\bar{n} = n/2$, die ein x_i, $i \in \{1, \ldots, \bar{n}\}$ enthalten und bezeichnen das System mit $\mathcal{D}'(z)$. Dabei wird n erstmal als gerade angenommen. Für die Definition der $\mathcal{D}_j$ sei auf Kapitel 5 verwiesen und eine mögliche Darstellung für 10 Beobachtungen wird in Abbildung 3 gezeigt.

Die Würfel aus $\mathcal{D}'(z)$ bilden einen Baum $\mathcal{T}'(\mathbf{z})$, den wir als *Belegungsbaum* bezeichnen. Vervollständigt man den Belegungsbaum $\mathcal{T}'(z)$, indem man zu allen Würfeln aus $\mathcal{D}'(z)$ deren Geschwister hinzufügt, so erhält man einen vollständigen Baum $\mathcal{T}(\mathbf{z})$, dessen Blätter eine Partition von $\mathbf{X}$ bilden, siehe dazu Abbildung 4 und Abbildung 5.

Ist $\mathcal{T}$ ein Teilbaum des Belegungsbaumes $\mathcal{T}(z)$ und vollständig, so definiert man

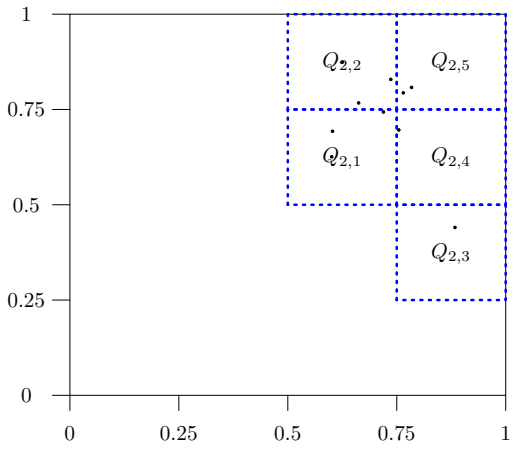
$$\gamma_{\mathcal{T}} := \sum_{Q \in \mathcal{L}(\mathcal{T})} \max(\bar{\eta}_Q, 0)$$

und nennt das die *Energie* des Baumes $\mathcal{T}$. Das bedeutet, dass der Mengenschätzer $\bar{\Omega}_m$ einem vollständigen Baum $\bar{\mathcal{T}}_m \in \mathfrak{T}_m$ entspricht, der die gerade definierte Energie maximiert. Es ist zu beachten, dass dieses Maximum von verschiedenen Bäumen angenommen werden kann. Da nur Werte $m \in \{1, \ldots, \bar{n}\}$ bei der Modellwahl betrachtet werden und ein Unterteilen von nicht belegten Würfeln keinen Energiezuwachs einbringt, ist ein maximierender Baum immer ein Teilbaum von $\mathcal{T}(\mathbf{z})$.

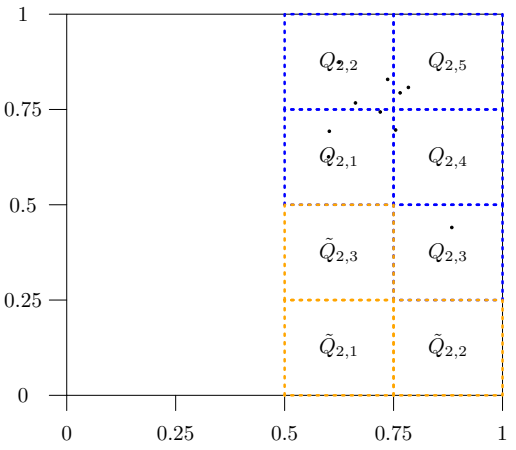


(a) Würfel aus $\mathcal{D}_1$.

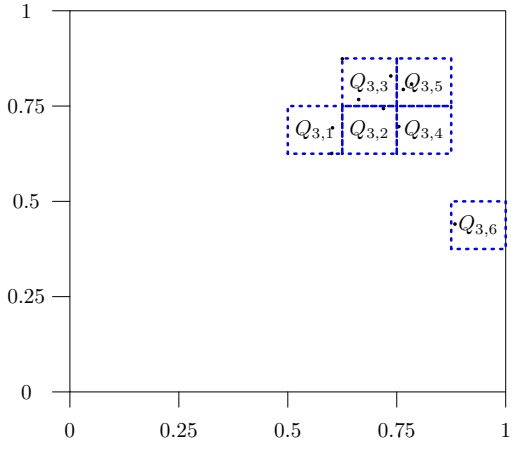
(b) Würfel aus $\mathcal{D}_1$ mit Geschwistern.



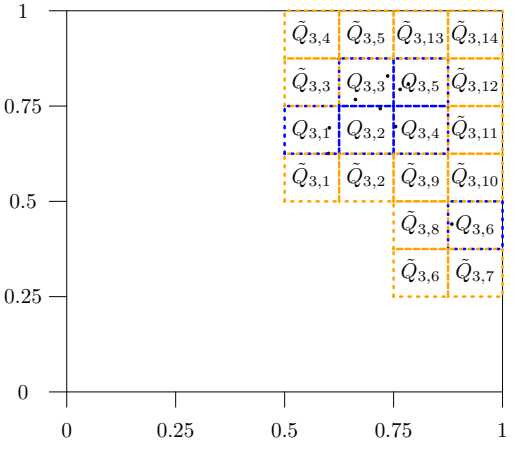
(c) Würfel aus $\mathcal{D}_2$.



(d) Würfel aus $\mathcal{D}_2$ mit Geschwistern.



(e) Würfel aus $\mathcal{D}_3$.



(f) Würfel aus $\mathcal{D}_3$.

Abbildung 3: Würfel und deren Geschwister, die zu einen Belegungsbaum mit 10 Beobachtungen gehören. Die blauen Umrandungen begrenzen die Bereiche der Würfel mit mindestens einer Beobachtung und die orangenen Umrandungen stellen die zugehörigen Geschwister dar.

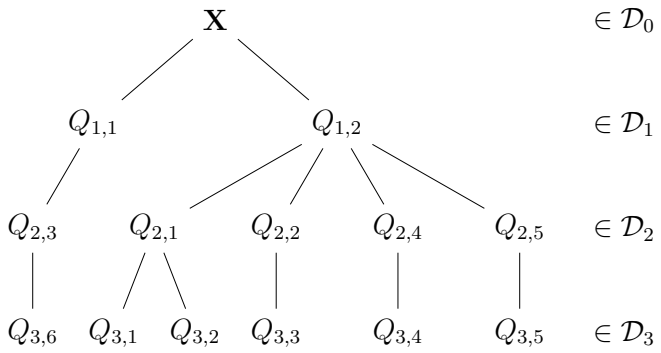


Abbildung 4: Belegungsbaum zu Abbildung 3 mit Höhe 4.

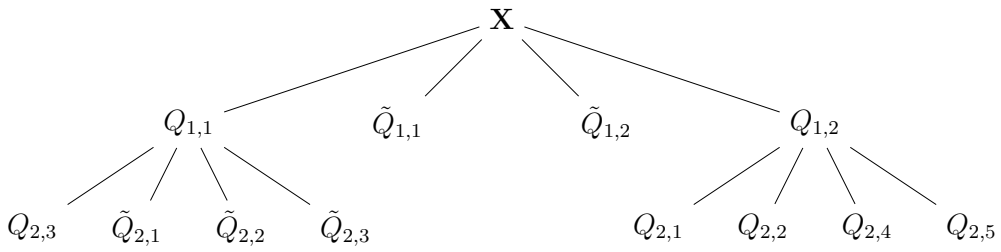


Abbildung 5: Vollständiger Belegungsbaum zu Abbildung 3 mit Höhe 3.

Ferner bezeichnen wir für jeden Würfel $Q \in \mathcal{T}(\mathbf{z})$ durch $\mathfrak{T}_m(Q)$ die Menge aller vollständigen Bäume $\mathcal{T}$ mit Wurzel Q, die man mittels höchstens m Unterteilungsschritten erhält und in $\mathcal{T}(\mathbf{z})$ enthalten sind. Damit definiert man

$$\gamma_{Q,m} := \max_{\mathcal{T} \in \mathfrak{T}_m(Q)} \gamma_{\mathcal{T}}.$$

Das Maximum kann auch hier durch unterschiedliche Bäume aus $\mathfrak{T}_m(Q)$ angenommen werden. Weiter bezeichnen wir mit $\mathcal{T}(Q,m)$ jeden Baum aus $\mathfrak{T}_m(Q)$ der das Maximum von $\gamma_{Q,m}$ annimmt. Wir vereinbaren

$$\mathcal{T}(Q,m) = \emptyset,$$

falls Q nicht belegt ist. Mit dieser Schreibweise definieren wir

$$\bar{\mathcal{T}}_m := \mathcal{T}(X,m) \qquad \text{und} \qquad \bar{\Omega}_m := \bigcup_{Q \in \mathcal{L}(\bar{\mathcal{T}}_m)} \{Q \colon \bar{\eta}_Q > 0\}. \tag{6.1}$$

Das benutzen wir zur vorher diskutierten Modellauswahl.

Hier wird nun ein Algorithmus beschrieben, der die vorherige Maximierung ausführt und dadurch $\bar{\mathcal{T}}_m$ liefert. Mit (6.1) erhalten wir daraus das benötigte $\bar{\Omega}_m$. Man beachte für nicht besetztes Q ist $\mathcal{T}(Q,m)$ leer und es gilt $\bar{\eta}_Q = \gamma_{Q,m} = 0$. Somit sind

diese Werte ohne Berechnung verfügbar. Deshalb finden Berechnungen nur für die besetzten Würfel statt, die $\mathcal{T}'(\mathbf{z})$ bilden. Für $l \in \{0, \dots, \bar{n}\}$ definieren wir noch

$$\mathcal{U}_l := \mathcal{T}'(\mathbf{z}) \cap \mathcal{D}_{\bar{n}-l},$$

die Menge der belegten Würfel des Rasterlevels $\bar{n} - l$. Dabei ist $\mathcal{U}_0 = \mathcal{L}(\mathcal{T}'(\mathbf{z}))$. Die Kalkulationen beginnen bei den Blättern von $\mathcal{T}'(\mathbf{z})$ und enden bei der Wurzel, ähnlich dem optimalen Zurückstutzen beim CART Algorithmus (siehe [Bre+84]). Der Ablauf wird in den folgenden Schritten beschrieben:

- $l = 0$: Man berechnet für jedes $Q \in \mathcal{U}_0$ die Größen $\bar{\eta}_Q$ und definiert $\gamma_{Q,0} := \max\{0, \bar{\eta}_Q\}$, sowie $\mathcal{T}(Q, 0) := \{Q\}$.

- $l \in \{1, \dots, \bar{n}\}$: Die restlichen Berechnungen bauen nun auf den vorher erhaltenen Ergebnissen auf. Nimmt man an, dass die Werte für $\gamma_{Q,j}, \bar{\eta}_Q$, sowie $\mathcal{T}(Q, j)$ für alle $Q \in \mathcal{U}_{l-1}$ mit $0 \leq j \leq l-1$ schon bestimmt sind, so lassen sich daraus die Werte für die Würfel in $\mathcal{U}_l$ berechnen. Für alle $0 \leq j \leq l-1$ und alle Würfel $Q \in \mathcal{U}_l$ ergibt sich der Vektor, der die Anzahl der Verfeinerungen der Nachfolgerwürfel ausrechnet und dabei die Energie der besten Teilbäume maximiert, wie folgt:

$$(l_j^*(R))_{R \in \mathcal{C}'(Q)} := \operatorname{argmax} \left\{ \sum_{R \in \mathcal{C}'(Q)} \gamma_{R,l'(R)} : \sum_{R \in \mathcal{C}'(Q)} l'(R) = j \right\}, \tag{6.2}$$

  hierbei bezeichnet $l'(R)$ die Anzahl der benutzten Verfeinerungen von R, die Menge $\mathcal{C}(Q)$ die Kinder von Q und ferner $\mathcal{C}'(Q) := \mathcal{C}(Q) \cap \mathcal{T}'(\mathbf{z})$ die Menge der nicht leeren Kinder von Q. Auch hier kann es mehrere Vektoren geben, die das Maximum aus (6.2) annehmen. Mit diesem Vektor lassen sich nun die restlichen Größen für jedes $Q \in \mathcal{U}_l$ bestimmen. Diese ergeben sich für jedes $1 \leq j \leq l$ durch

$$\gamma_{Q,j} = \sum_{R \in \mathcal{C}'(Q)} \gamma_{R,l_{j-1}^*(R)},$$

  und

$$\mathcal{T}(Q, j) = \{Q\} \cup \left(\bigcup_{R \in \mathcal{C}'(Q)} \mathcal{T}(R, l_{j-1}^*(R)) \right) \cup (\mathcal{C}(Q) \backslash \mathcal{C}'(Q)) .$$

  Für $j = 0$ berechnet sich $\bar{\eta}_Q$ für alle $Q \in \mathcal{U}_l$ durch Aufsummieren der $\bar{\eta}_R$ für $R \in \mathcal{C}'(Q)$ und man definiert wie im Fall $l = 0$ die Größen $\gamma_{Q,0} := \max\{0, \bar{\eta}_Q\}$ und $\mathcal{T}(Q, 0) = \{Q\}$.

```
265 labeledTree calculateAdaptiveDyadicTree(
266     NumericMatrix& X1,
267     NumericMatrix& X2,
268     const IntegerVector& Y1,
269     const IntegerVector& Y2,
270     unsigned int max_depth,
271     unsigned int max_subdivisions,
272     unsigned int distinctLabels,
273     unsigned int emptyLeafLabel)
```

Programmauszug 6.1: Parameter der Hauptfunktion. CUDTP.cpp

Nach Ausführen des letzten Schrittes $l = \bar{n}$ besteht die Menge $\mathcal{U}_{\bar{n}}$ nur noch aus der Wurzel $\mathbf{X}$ und somit hat man die Größen $\mathcal{T}(\mathbf{X}, m)$ für $m = 0, \ldots, \bar{n}$ bestimmt. Deshalb stehen einem nun auch die Schätzer $\bar{\Omega}_m$ für $m = 0, \ldots, \bar{n}$ zur Verfügung.

Bei der Berechnung der kleinsten Zellen fällt auf, dass schon für relativ kleine Werte von $\bar{n}$ die Vektoren der Indizes, die eine Zelle eindeutig bestimmen, sehr groß werden können. Zum Beispiel für $\bar{n} = 100$ gibt es dann 2^{100} mögliche Werte für ein Element dieses Vektors, wenn man sich das zugehörige Gitter vorstellt. Das überschreitet die verfügbare Anzahl von Zahlen der Typen *unsigned int* und *unsigned long* in C++. Deswegen wurde hier die Bibliothek GMP benutzt, die unter anderem einen Datentyp für große Ganzzahlen *mpz_t* und die zugehörige Klasse *mpz_class* zur Verfügung stellt. Dadurch kann es jedoch zu Geschwindigkeitseinbußen im Vergleich zu den Standardtypen kommen.

Daher wurden verschiedene Parameter eingeführt, die das Modell beschränken können. Der Parameter *max_depth* wird dabei verwendet, um die Baumtiefe einzuschränken. Ferner wird die maximale Anzahl an Unterteilungen, die ein Baum haben kann, durch *max_subdivisions* reguliert. Somit stehen bei der Modellwahl genau *max_subdivisions* Bäume zur Verfügung. Effektiv ist dieser Parameter auch eine Beschränkung der Tiefe, da sie ebenfalls nicht über diesen Wert hinausgehen kann und die tatsächlich mögliche Tiefe ist dann das Minimum aus beiden Werten. Da es wünschenswert sein kann mehr Trainingsbeobachtungen zur Erstellung der Bäume auf Kosten des Vergleichs unter den Bäumen zu haben, gibt es noch die Variable *nbar*, die angibt wie viele Beobachtungen zur Konstruktion der Bäume verwendet werden. Entsprechend bleiben zum Vergleich der Bäume bei der Modellwahl noch $n - nbar$ Beobachtungen. Die Aufteilung erfolgt dabei schon in R, sodass an die Hauptfunktion zwei Matrizen für die Prädiktorvariablen und zwei Vektoren für die Zielmerkmale übergeben werden, siehe Programmauszug 6.1.

```
418       vector<unsigned int> children_m_sub_vector =
      children_max_subdivisions(egBegin,n_children);
421       size_t max_possible_subdivisions_child = *min_element(
      children_m_sub_vector.begin(),children_m_sub_vector.end());
423       size_t loopend = l+max_subdivisions-needed_depth;
424       for(size_t j = 1; j<=loopend;j++)
425       {
429         if(loopend - max_possible_subdivisions_child == 1)
430         {
431           l_star_vec = argmax_l_star(egBegin,j-1,n_children);
432         }
433         else
434         {
436           l_star_vec = argmax_l_star(egBegin,j-1,children_m_sub_vector,
      n_children);
437           if(l_star_vec.size() == 0)
438             break;
439         }
441         (*egtNewIt)[index].gamma.push_back(sum_energie(
442             egBegin,
443             n_children,
444             l_star_vec
445         ));
448         (*egtNewIt)[index].tree.push_back(mergeTrees(
449             egBegin,
450             uBegin,
451             l_star_vec
452         ));
453       }
```

Programmauszug 6.2: Berechne den Vektor l_j^*. CUDTP.cpp

Die Variable *distinctLabels* gibt die Anzahl der unterschiedlichen Klassen an und ist im Standardfall gleich 2, kann aber auch 1 sein. Im dyadischen Baum bestimmt sie für wie viele Klassen die Anzahl der zugehörigen Beobachtungen in einem Blatt gezählt wird. Für den Fall einer leeren Zelle gibt es die Variable *emptyLeafLabel*, die die Klasse angibt, welche zugeordnet werden soll.

Durch die neuen Parameter muss der Algorithmus leicht abgewandelt werden. Zum Beispiel nimmt die Hauptschleife für l nicht mehr Werte bis $\bar{n}$, sondern nur noch Werte bis *needed_depth* an. Dabei ist *needed_depth* das Minimum von *max_depth* und *max_subdivisions*.

Die größte Veränderung tritt bei der Berechnung des Verfeinerungsvektors l_j^* aus (6.2) ein. Zum einen wird in Programmauszug 6.2 bestimmt, wie viele Unterteilungen die Nachfolgerzellen haben. Das ist die maximale Anzahl an Unterteilungen dieser Zellen. Weiter wird festgelegt, wann die Schleife spätestens abgebrochen werden muss, weil nicht mehr Unterteilungen für die betrachtete Zelle möglich sind.

Innerhalb der Schleife wird eine Fallunterscheidung getroffen, ob der normale Algorithmus zur Berechnung von l_j^* genutzt wird oder ob die abgewandelte Form mit unterschiedlicher maximaler Anzahl an Unterteilungen der Nachfolgerzellen zum Einsatz kommt. Bei der abgewandelten Form, kann es passieren, dass der zurückgegebene Vektor die Länge 0 besitzt. Das bedeutet, dass beim Suchen des Argumentes des Maximums kein Vektor gefunden werden konnte, weil es für das entsprechende j keinen solchen Vektor gibt. Deswegen wird die Schleife abgebrochen.

Für das Durchlaufen der Argumente in (6.2) wurde Algorithmus 1 aus [Knu05] verwendet, der dort mit Algorithmus L (Lexikographische Kombinationen) bezeichnet wird.

Algorithmus 1 : Lexikographische Kombinationen

Beschreibung : Der Algorithmus generiert alle t-Kombinationen $c_t \dots c_2 c_1$ der n Zahlen $\{0, 1, \dots, n-1\}$, falls $n \leq t \leq 0$ gilt. Dabei werden die zusätzlichen Variablen c_{t+1} und c_{t+2} als Markierungen benutzt.

L1. [Initialisiere.] Setze $c_j \leftarrow j - 1$ für $1 \leq j \leq t$, sowie $c_{t+1} \leftarrow n$ und $c_{t+2} \leftarrow 0$.

L2. [Besuche.] Besuche die Kombination $c_t \dots c_2 c_1$.

L3. [Finde j.] Setze $j \leftarrow 1$. Solange $c_j + 1 = c_{j+1}$ ist, setze $c_j \leftarrow j - 1$ und $j \leftarrow j + 1$.

L4. [Fertig?] Beende den Algorithmus wenn $j > t$.

L5. [Erhöhe c_j.] Setze $c_j \leftarrow c_j + 1$ und gehe zurück zu L2.

In der Implementierung ist dann Schritt L2. aus Algorithmus 1 wichtig, der sich für die verschiedenen Fälle in Programmauszug 6.2 auch unterscheidet. Anstatt der t-Kombinationen $c_t \dots c_2 c_1$ benötigen wir die Kombinationsvariante aus [Knu05] mit den q_i, die in der Summe j ergeben. Also werden in Schritt L2. die c_i in q_i umgewandelt und für den zugehörigen l_j^* Vektor geschaut, ob er bzgl. der Energie maximal ist. Ist er maximal, so wird er abgespeichert und im anderen Fall verworfen, siehe dazu Programmauszug 6.3.

Für den Spezialfall variabler Maximalanzahl der Unterteilungen der Zellen werden zu viele Konfigurationen durchlaufen, vgl. Programmauszug 6.4. Das ist ein guter Ansatzpunkt, um die Implementierung zu optimieren.

```
146     for(size_t ind = 0; ind != size; ind++)
147     {
148       q_k[ind] = (cs[ind+1]-cs[ind])-1;
149     }
150     new_gamma = sum_energie(begin,size,q_k);
151     if(max_gamma < new_gamma)
152     {
153       max_int_vec = q_k;
154       max_gamma = new_gamma;
155     }
```

Programmauszug 6.3: Durchlaufe alle möglichen Kombinationen für l_j^*. CUDTP.cpp

```
198     l_star_out_of_range = false;
199     for(size_t ind = 0; ind != size; ind++)
200     {
201       q_k[ind] = (cs[ind+1]-cs[ind])-1;
202       if(q_k[ind] > max_values[ind])
203       {
204         l_star_out_of_range = true;
205         break;
206       }
207     }
209     if(!l_star_out_of_range)
210     {
211       new_gamma = sum_energie(begin,size,q_k);
212       if(max_gamma < new_gamma)
213       {
214         max_int_vec = q_k;
215         max_gamma = new_gamma;
216       }
217     }
```

Programmauszug 6.4: Durchlaufe alle möglichen Kombinationen für l_j^* bei variabler Maximalanzahl der Unterteilungen der Zellen. CUDTP.cpp

6.2 Optimal Dyadic Decision Tree (ODT)

ODT steht für „Optimal dyadic decision tree“. Die Idee des Algorithmus entstammt aus [Bla+07, S. 3 ff.]:

Hier wird ein geeigneter Baum durch empirisch bestrafende Kostenminimierung ausgewählt. In diesem Algorithmus wird eine dyadische Partition als eine Aufteilung des Hyperwürfels $[0;1]^d$ definiert. Das geschieht durch Zerteilen rechtwinklig zu einer der Koordinatenachsen durch den jeweiligen Mittelpunkt und dann rekursivem Aufspalten in wieder gleiche Hälften, bis man an einem frei wählbaren Punkt eines Zweiges angekommen ist. Dadurch ist jedes Teil der Partition ein dyadisches Parallelepiped, also ein kartesisches Produkt von Intervallen der Form $\left[\frac{i}{2^k};\frac{(i+1)}{2^k}\right)$. Im Folgenden wird dies Zelle genannt. Es gibt keine vorgeschriebene Reihenfolge, in

der die Teilung erfolgt und insbesondere kann eine Koordinate mehrere Male hintereinander für die Aufspaltung genutzt werden. Die Konstruktion lässt sich als Baum veranschaulichen.

Um die Zellen nicht zu fein werden zu lassen und die Kalkulationen einzuschränken, wird ein Parameter k_{max} eingeführt, der die Anzahl der Spaltungen in eine Richtung begrenzt. Da es d Richtungen gibt, ist die maximale Tiefe eines Baumes $d \cdot k_{max}$. Bezeichne mit $\mathfrak{B}_{k_{max}}$ die Menge der dyadischen Partitionen, die diese Eigenschaft erfüllen. Dabei darf k_{max} nicht von der Realisation der Beobachtungsdaten abhängen, jedoch kann es das von der Anzahl der Beobachtungen.

Sei $\mathcal{B}$ eine solche dyadische Partition, dann wird ein Häufigkeitsschätzer durch

$$\forall b \in \mathcal{B}\ \forall x \in b \qquad \hat{f}_{\mathcal{B}}(x, y) := \frac{N_{b,y}}{\sum\limits_{y} N_{b,y}}$$

definiert. Wenn S die Anzahl der Klassen ist, so gilt $y \in \{1, \ldots, S\}$. $N_{b,y}$ bezeichnet hierbei die Anzahl der Trainingspunkte der Klasse y, die in Zelle b liegen. Zur Klassifikation wird dann die Klasse verwendet, die in der Zelle am häufigsten vorkam.

Die Modellwahl hängt von einem Kriterium ab, das misst wie gut ein Schätzer ein bestimmtes Ziel erfüllt. Dazu wird eine Kostenfunktion $l(f, x, y) \in \mathbb{R}$ verwendet, die im Mittel so gering wie möglich sein soll. Unsere Kostenfunktion wird die Anzahl der falsch klassifizierten Beobachtungen sein:

$$l_{class}(f, x, y) = \mathbb{1}_{\{f(x) \neq y\}}.$$

Deswegen ergibt sich unser Klassifikator durch empirische Klassifikationsverlustminimierung. Wir wählen die Partition dann entsprechend

$$\hat{\mathcal{B}} = \operatorname*{argmin}_{\mathcal{B} \in \mathfrak{B}_{k_{max}}} \frac{1}{n} \sum_{i=1}^{n} l(\hat{f}_{\mathcal{B}}, x_i, y_i) + \gamma |\mathcal{B}|,$$

wobei $|\mathcal{B}|$ die Anzahl der Elemente der Partition $\mathcal{B}$ bezeichnet. Das sind die Blätter des dyadischen Baumes, der die Partition erzeugt hat. γ ist eine regularisierende Konstante, die größer als eine Funktion der Form $\frac{c}{n}$ oder $\frac{c \cdot \ln(n)}{n}$, je nach Rahmenbedingung, sein sollte. Damit erfolgt eine angepasste Wahl in dem Sinne, dass ein automatischer Ausgleich zwischen Schätz- und Näherungsfehler erfolgt.

Der Algorithmus baut auf dem Prinzip auf, dass die zu optimierende Funktion additiv über die Teile der Partition ist.

$$\frac{1}{n} \sum_{i=1}^{n} l(\hat{f}_{\mathcal{B}}, x_i, y_i) + \gamma |\mathcal{B}| = \sum_{b \in \mathcal{B}} \left(\gamma + \frac{1}{n} \sum_{i:\ x_i \in \mathcal{B}} l(\hat{f}_b, x_i, y_i) \right) =: \sum_{b \in \mathcal{B}} \mathcal{E}(\{b\}) \qquad (6.3)$$

mit $\hat{f}_b$ als konstanter Wert von $\hat{f}_{\mathcal{B}}$ auf der Zelle b und $\mathcal{E}(\{b\})$ als implizit definierter Kostenfunktion eingeschränkt auf die Zelle b. Die Tiefe einer Zelle wird dabei als Anzahl der Schnitte definiert, die nötig sind, um die Zelle zu erhalten. Das entspricht der Tiefe einer Zelle in einem dyadischen Entscheidungsbaum.

Kennt man die optimale Partition für alle Zellen einer gewissen Tiefe k, so lässt sich daraus leicht die optimale Partition für jede Zelle der Tiefe $k-1$ berechnen, da man nur die obige additive Eigenschaft ausnutzen muss. Die Bestimmung der optimalen Partitionen für alle Zellen und alle Tiefen führt schnell zu eine kombinatorischen Explosion. Es gibt bereits $2^{d \cdot k_{max}}$ kleinste Zellen und für Zwischentiefen ergeben sich sogar noch mehr Zellen. Jedoch enthalten viele dieser Zellen keine Trainingspunkte, da es mehr Zellen als Beobachtungen gibt. Für leere Zellen ist die optimale Partition, die Zelle selbst. Deshalb muss man nur Informationen zu den nichtleeren Zellen speichern. Dies geschieht mit Lexika $\mathcal{D}_k$, die die Zelle b, deren Tiefe k und die optimale Partition T_b^* abspeichern. Man nutzt zur Erstellung von $\mathcal{D}_{k-1}$ nur Zellen aus $\mathcal{D}_k$. Am Ende ergibt sich in $\mathcal{D}_0$ die Partition, die das Optimierungsproblem (6.3) minimiert. Der Ablauf wird in Algorithmus 2 zusammengefasst.

Algorithmus 2 : Lexikon-basierender ODT Algorithmus

L1. **Initialisierung**: Konstruiere das Lexikon $\mathcal{D}_{d \cdot k_{max}}$:
L2. **für** $i = 1, \ldots, n$ **tue**
L3. Finde für die Beobachtung x_i die kleinste Zelle b_i (den Hyperwürfel der Kantenlänge $2^{-k_{max}}$), die x_i enthält und speichere sie in $\mathcal{D}_{d \cdot k_{max}}$ zusammen mit der trivialen Partition $T_{b_i}^* = \{b_i\}$.
L4. **Hauptteil**:
L5. **für** $D = d \cdot k_{max}, \ldots, 1$ **tue**
L6. Initialisiere $\mathcal{D}_{D-1} = \emptyset$. ;
L7. **für** $b \in \mathcal{D}_D$ **tue**
L8. **für** $k = 1, \ldots, d$ **tue**
L9. **wenn** b *hat einen Geschwisterteil entlang der Dimension* k **dann**
L10. Bezeichne ihn mit b' und schlage die Zelle im Lexikon $\mathcal{D}_D$ nach. Falls sie nicht gefunden wird, so setze die optimale Partition $T_{b'}^* = \{b'\}$.
L11. Bezeichne u als die direkte Vorgängerzelle von b und b' ($u = b \cup b'$).
L12. **wenn** u *ist bereits in* $\mathcal{D}_{D-1}$ *mit* T_u^* *abgespeichert* **dann**
L13. $T_u^* \leftarrow \operatorname{argmin} \{\mathcal{E}(T_u^*), \mathcal{E}(T_b^* \cup T_{b'}^*) = \mathcal{E}(T_b^*) + \mathcal{E}(T_{b'}^*)\}$
L14. **sonst**
L15. $T_u^* \leftarrow \operatorname{argmin} \{\mathcal{E}(\{u\}), \mathcal{E}(T_b^* \cup T_{b'}^*) = \mathcal{E}(T_b^*) + \mathcal{E}(T_{b'}^*)\}$

```
298   for(int index = 0; index < n;index++)
299   {
300     minCells.at(index) = getMinimalCell(doubleVector(Xvector(Range(0,n
      -1),Range(0,d-1)),index),d,kmax);
301     auto iter = cellContains.find(size_tVectorkmax(minCells[index],kmax
      ));
302     if(iter == cellContains.end())
303     {
304       cellContains.insert( sizetpair(
305         size_tVectorkmax(minCells[index],kmax),
306         size_tVector(vector<size_t>(1,index)) ));
307     }
308     else
309     {
310       iter->second.v.push_back(index);
311     }
312   }
```

Programmauszug 6.5: Generierung und Speicherung der Zellen. ODT.cpp

Hier folgen jetzt einige Erklärungen, wie Algorithmus 2 umgesetzt wurde. Die Hauptfunktion ist in C++ geschrieben, um Geschwindigkeitsvorteile zu erreichen und den Speicher selbst zu verwalten. Sie benötigt als Parameter die Anzahl der Trainingsbeobachtungen n, die Anzahl der Prädiktorvariablen d, die Anzahl der Spaltungen die in eine Richtung maximal möglich sind k_{max}, die Anzahl verschiedener Klassen S, den Wert für die Regularisierungskonstante γ, die Prädiktorvariablen als Matrix, wobei die Zeilen Beobachtungen darstellen und die zugehörigen Zielmerkmale als Vektor. Zurückgegeben wird dabei eine Liste, die die optimale Partition darstellt. Jedes Element dieser Liste enthält dabei Informationen zur Begrenzung der Zelle, wie häufig eine Klasse in der Zelle vorkommt und die berechneten Kosten der Zelle. In R wird die Liste dann in eine Liste, die einen Baum nachbildet, umgewandelt. Diese Umwandlung kann noch optimiert werden, indem auch das in C++ geschieht bzw. auch die Vorhersage in C++ berechnet wird.

Es werden verschiedene Zellklassen definiert, die einen unterschiedlichen Anteil an Informationen über diese Zelle enthalten und als Lexika werden Objekte der Klasse std::unordered_map verwendet. Dafür benötigt man die Version C++11.

In Programmauszug 6.5 wird für jede Beobachtung die kleinste Zelle berechnet, in der die Beobachtung enthalten ist und in der Variable *minCells* abgespeichert. Im nächsten Schritt wird in *cellContains* zu jeder nicht leeren Zelle abgespeichert, welche Beobachtungen enthalten sind.

Der zweite Teil der Initialisierung des ODT Algorithmus erfolgt in Programmauszug 6.6. Es werden in *trivialCells* die trivialen Partitionen abgespeichert, auf die später mittels Zeigern zurückgegriffen wird und in *dictionary*[0] werden dann Par-

```
313   auto trivOldIt = trivialCells.insert(trivialCells.begin(),cubmap());
314   dictionary[0] = partitionmap();
315   auto dictionaryIt = &(dictionary[0]);
316   for(auto iter = cellContains.begin(); iter != cellContains.end();
        iter++)
317   {
318     size_tVector newNby(S);
319     for(auto iterat = iter->second.v.begin(); iterat != iter->second.v.
      end();iterat++)
320     {
322       newNby[Yvector(*iterat)-1]++;
323     }
324     auto tcpairIterator = trivOldIt->insert(trivOldIt->begin(),cubpair(
325       cuboidCellkmax(iter->first),
326       cuboidExt(newNby,0) ));
327     auto dicpairIterator = dictionaryIt->insert(dictionaryIt->begin(),
      partitionpair(
328       cuboidCellkmax(iter->first),
329       cuboidPartition(cuboidCell(iter->first.vec,iter->first.vec),
      tcpairIterator)));
330     tcpairIterator->second.cost = costtrivial(dicpairIterator->second,d
      ,kmax);
331   }
```

Programmauszug 6.6: Speichere die trivialen Partitionen ab. ODT.cpp

titionen abgespeichert, die aus den trivialen Partitionen bestehen. Dadurch werden nicht $d \cdot k_{max} + 1$ Lexika benötigt, sondern nur 2 für die gerade benutzten Partitionen der Stufe D und $D-1$, sowie ein Lexikon für die trivialen Partitionen, der jeweiligen Stufe.

Das suchen des Geschwisterteils zur Zelle b erfolgt in Programmauszug 6.7. Dabei wird geschaut, ob die Zelle bei Verdoppelung in Richtung k noch innerhalb des Hyperwürfels $[0;1]^d$ liegt. Ist das der Fall, so hat b einen Geschwisterteil und er muss noch im Lexikon gefunden werden. Ist er nicht im Lexikon vorhanden, so muss ein neues Element zu den trivialen Partitionen hinzugefügt werden. Danach wird die Partition und die Begrenzung der Vorgängerzelle von b und b' zwischengespeichert.

Zuletzt werden in Programmauszug 6.8 noch die provisorischen Teilpartitionen abgespeichert und vorher sich eventuell neu ergebende triviale Partitionen erzeugt, die ins zugehörige Lexikon eingetragen werden. Zusätzlich werden die Zellen der Partition nach Größe der Indizes geordnet, um die Erstellung des dyadischen Baumes zu erleichtern.

```
343         if(hasSibling(pairIterator->first.cell,k,kmax))
344         {
345           cuboidCell bSibling = getSibling(pairIterator->first.cell,k,
    kmax);
346           cuboidPartition TbSibling;
347           auto bSiblingIt = dicOldIt->find(cuboidCellkmax(bSibling,kmax
    ));
348           if(dicOldIt->end()==bSiblingIt)
349           {
350             auto trivPairIt = trivOldIt->find(cuboidCellkmax(bSibling,
    kmax));
351             if(trivPairIt == trivOldIt->end())
352             {
353               (*trivOldIt)[cuboidCellkmax(bSibling,kmax)] = cuboidExt(
    size_tVector(S),gamma);
354               trivPairIt = trivOldIt->find(cuboidCellkmax(bSibling,kmax
    ));
355             }
356             TbSibling = cuboidPartition(bSibling,trivPairIt);
357           }
358           else
359           {
360             TbSibling = bSiblingIt->second;
361           }
362           cuboidCell bParent = getParent(pairIterator->first.cell,k,
    kmax);
363           cuboidCellkmax bParentkmax = cuboidCellkmax(bParent,kmax);
```

Programmauszug 6.7: Finde Geschwister- und Vorgängerzellen. ODT.cpp

```
364           auto pairIt = dicNewIt->find(bParentkmax);
365           if(pairIt == dicNewIt->end())
366           {
367             auto trivPairIt = trivNewIt->find(bParentkmax);
368             if(trivPairIt == trivNewIt->end())
369             {
370               cuboidCellExt trivPartition;
371               if(pairIterator->second.cell.lowIndices < TbSibling.cell.
    lowIndices)
372                 trivPartition = createPartitionFromParts(pairIterator->
    second,TbSibling);
373               else
374                 trivPartition = createPartitionFromParts(TbSibling,
    pairIterator->second);
375               trivPairIt = trivNewIt->insert(trivNewIt->begin(),
    make_pair(bParentkmax,trivPartition.ext));
376             }
377             cuboidPartition p1 = cuboidPartition(bParent,trivPairIt),p2
    ;
379             if(pairIterator->second.cell.lowIndices < TbSibling.cell.
    lowIndices)
380             {
381               p2 = cuboidPartition(pairIterator->second,TbSibling);
382             }
383             else
384             {
385               p2 = cuboidPartition(TbSibling,pairIterator->second);
386             }
387             dicNewIt->insert(dicNewIt->begin(),make_pair(
388               bParentkmax,
389               ArgMin<const cuboidPartition&,double>(costfunction,p1,p2)
    ));
390           }
391           else
392           {
394             if(pairIterator->second.cell.lowIndices < TbSibling.cell.
    lowIndices)
395             {
396               cuboidPartition p2 = cuboidPartition(pairIterator->second
    ,TbSibling);
397               pairIt->second = ArgMin<const cuboidPartition&,double>(
    costfunction,pairIt->second,p2);
398             }
399             else
400             {
401               cuboidPartition p2 = cuboidPartition(TbSibling,
    pairIterator->second);
402               pairIt->second = ArgMin<const cuboidPartition&,double>(
    costfunction,pairIt->second,p2);
403             }
404           }
```

Programmauszug 6.8: Bestimme optimale Teilpartitionen. ODT.cpp

7 Vergleich mit typischen Verfahren zur Klassifizierung

7.1 Typische Verfahren

Dieser Abschnitt soll kurz in einige typische Klassifizierungsverfahren des statistischen Lernens einführen und mathematische Hintergründe, sowie Material zum Weiterlesen liefern. Hierzu sei auf [Jam+16] und [HTF09] verwiesen, aus dem ein Großteil des folgenden Materials stammt. Außerdem wird auf die Bezeichnungen und Begriffe aus Kapitel 2 zurückgegriffen.

7.1.1 Lineare Regression

Die lineare Regression ist ein sehr einfacher Ansatz zum überwachten Lernen. Insbesondere wird dieser Ansatz für die Vorhersage von quantitativen Werten benutzt. Diese Art der Regression ist schon seit Längerem bekannt und wird in zahlreichen Büchern behandelt. Obwohl diese Methode im Vergleich zu modernen statistischen Verfahren etwas stumpf wirkt, ist sie trotzdem eine nützliche und weitverbreitete statistische Lernmethode. Darüber hinaus bauen viele der neueren Lernstrategien auf linearer Regression auf und bilden Verallgemeinerungen oder Erweiterungen dessen. Einige davon werden auch zur Klassifizierung benutzt. Somit ist das Verständnis der linearen Regression und ihrer Ideen ein guter Ausgangspunkt, um auch komplexere Ansätze zu erfassen.

Die einfachste Art ist die einfache lineare Regression. Dabei wird ein numerisches Zielmerkmal anhand eines einzelnen Prädiktors vorhergesagt. Für das zugehörige Modell wird also angenommen, dass ein linearer Zusammenhang zwischen X und Y besteht. Es hat die Form:

$$Y \approx \beta_0 + \beta_1 X. \tag{7.1}$$

Dabei bedeute "$\approx$" soviel wie "wird näherungsweise modelliert durch", wobei β_0 und β_1 unbekannte, reelle Konstanten sind. Sie werden auch als die Koeffizienten bzw. die Parameter des linearen Modells bezeichnet. Hat man mit den Beobachtungsdaten Schätzungen $\hat{\beta}_0$ und $\hat{\beta}_1$ erhalten, so ergibt sich eine Vorhersage $\hat{y}$ für den Wert y zum zugehörigen Wert x wie folgt:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x. \tag{7.2}$$

Nun stellt sich die Frage, wie man Schätzungen für die Parameter erhält und welche Kriterien diese erfüllen sollen. Dabei versucht man die Koeffizienten so zu wählen, dass die Trainingspunkte so nah wie möglich an der Linie aus (7.2) liegen. Dazu muss man jedoch definieren, was in diesem Zusammenhang Nähe bedeutet und wie man das messen kann. Es gibt viele verschiedene Ansätze dafür. Typischerweise benutzt man die Methode der kleinsten Quadrate. Dabei werden die Koeffizienten so gewählt, dass die Restsumme der Quadrate (RQ) minimiert wird:

$$RQ = \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i \right)^2,$$

vgl. hierzu [Jam+16, S.59 f.]. Als Beispiel wird in Abbildung 6 die lineare Regression mittels kleinster Quadrate für die Airpassenger Daten in R durchgeführt.

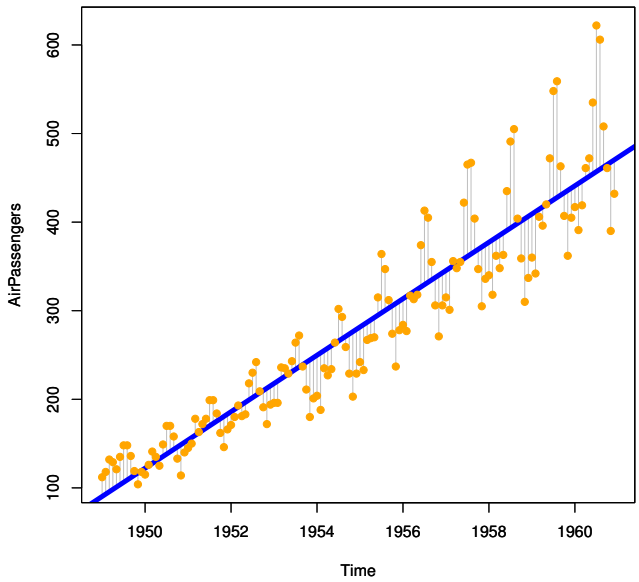


Abbildung 6: Lineare Regression Airpassengerdatensatz in R.

Im Allgemeinen Fall hat man d Prädiktoren $X_1, \ldots, X_d$ und erhält analog zu (7.1) die Form:

$$Y \approx \beta_0 + \sum_{i=1}^{d} \beta_i X_i,$$

mit unbekannten Konstanten $\beta = (\beta_0, \ldots, \beta_d)$. Weiter ergibt sich dann entsprechend zu (7.2)

$$\hat{y} = \hat{\beta}_0 + \sum_{i=1}^{d} \hat{\beta}_i x^{(i)},$$

mit den Schätzparametern $\hat{\beta} = (\hat{\beta}_0, \ldots, \hat{\beta}_d)$, wobei $x^{(i)}$ die i-te Komponente von x bezeichnet. Für die Methode der kleinsten Quadrate ergibt sich

$$RQ = \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{k=1}^{d} \hat{\beta}_k x_i^{(d)} \right)^2 .$$

Bezeichne mit $\mathbb{X}$ die $n \times (d+1)$ Matrix bestehend aus den Beobachtungsdaten, wobei die erste Spalte nur aus Einsen besteht und die restlichen Spalten die d Komponenten beinhaltet. Die n Beobachtungen bilden dabei die Zeilen. Dann erhält man unter der Annahme, dass $\mathbb{X}$ von vollem Rang ist, die eindeutige Lösung für das Minimierungsproblem durch:

$$\hat{\beta} = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbb{Y},$$

mit dem Spaltenvektor $\mathbb{Y}$, der aus den y-Werten der n Beobachtungen besteht. Ist $\mathbb{X}$ nicht von vollem Rang, so ist $\hat{\beta}$ nicht eindeutig bestimmt und man versucht die Matrix $\mathbb{X}$ zu reduzieren. Die meisten Software Pakete erkennen das Problem und können damit umgehen, siehe dazu auch [HTF09, S.44 ff.].

Da die lineare Regression meist eine starke Vereinfachung der Wirklichkeit darstellt, können eine Vielzahl von Problemen bei der Vorhersage auftreten. Nach [Jam+16, S.92 ff.] gehören dabei Ausreißer in den Daten, nichtlineare Beziehung zwischen Zielmerkmal und Prädiktoren, sowie Kollinearität bei den unabhängigen Variablen zu den häufigsten Schwierigkeiten.

Bei kategorischen Zielmerkmalen hat man zusätzliche Schwierigkeiten. Zuerst muss man den Klassen Werte zuweisen. Jedoch hat man in den meisten Fällen keine natürliche Ordnung dieser Klassen. Daraus ergibt sich für Zielmerkmale mit mehr als zwei Klassen, je nach Zuweisung mehrere völlig verschiedene lineare Modelle und damit stark abweichende Vorhersagen. Liegen nur zwei Klassen vor, so kann man versuchen, über eine Pseudovariable $\tilde{Y}$ die Klassifizierung vorzunehmen. Eine Klasse erhält den Wert 0, die andere den Wert 1. Nun führt man eine normale lineare Regression durch und für Werte größer als 0,5 weist man die Klasse mit dem Wert 1 zu und sonst die andere Klasse. Jedoch können die Werte, die die Pseudovariable $\tilde{Y}$ annimmt, auch außerhalb des Intervalls $[0; 1]$ liegen. Somit ist $\tilde{Y}$ nicht als Wahrscheinlichkeit interpretierbar. Deswegen ist die lineare Regression zur Klassifizierung nur schlecht geeignet.

Um die hier genannten Probleme zu lösen, wurden Ansätze wie logistische Regression und lineare Diskriminanzanalyse entwickelt, siehe auch [Jam+16, S.129 ff.].

7.1.2 Logistische Regression (Logit)

Bei der logistischen Regression versucht man die a-posteriori Wahrscheinlichkeiten $\mathbf{P}[Y = k|X = x]$ zu modellieren. Im Fall von zwei Klassen $\{1, 2\}$ wird häufig das Modell

$$\begin{aligned} \mathbf{P}[Y = 1|X = x] &= \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)}, \\ \mathbf{P}[Y = 2|X = x] &= \frac{1}{1 + \exp(\beta_0 + \beta^T x)} \end{aligned} \tag{7.3}$$

angewendet. Dabei bezeichne $\beta_0, \beta = (\beta_1, \ldots, \beta_d)$ wieder, wie auch bei der linearen Regression Abschnitt 7.1.1, die Parameter des Modells.

Wendet man nun auf die erste Wahrscheinlichkeit von (7.3) die sogenannte Logit Transformation $\ln[p/(1-p)]$ an, so ergibt sich

$$\ln\left(\frac{\mathbf{P}[Y = 1|X = x]}{\mathbf{P}[Y = 2|X = x]}\right) = \beta_0 + \beta^T x. \tag{7.4}$$

Die linke Seite von (7.4) nennt man auch Log-Odds oder Logit. Bei der Klassifikation wird man im Normalfall die Klasse zuweisen, die eine höhere Wahrscheinlichkeit besitzt. Da hier nur zwei Klassen eine Rolle spielen, ist für ein spezielles x die Wahrscheinlichkeit der einen Klasse die Gegenwahrscheinlichkeit der anderen Klasse. Somit spielt hier die Wahrscheinlichkeit 0,5 eine besondere Rolle, weil an dieser Stelle beide Wahrscheinlichkeiten gleich groß sind. In (7.4) eingesetzt, ergibt das als Klassengrenze eine Hyperebene, die durch $\{x \mid \beta_0 + \beta^T x = 0\}$ definiert wird. Es ergeben sich also lineare Klassengrenzen. Damit fällt die logistische Regression in den Bereich der linearen Methoden, vgl. hierzu [HTF09, S. 101 f.].

Um die Parameter des Modells zu bestimmen, wird die Maximum-Likelihood-Methode benutzt. Auch könnte man wieder die Strategie der kleinsten Quadrate im nichtlinearen Fall anwenden, jedoch besitzt sie in diesem Fall nicht so gute statistische Eigenschaften wie Maximum-Likelihood. Hier soll nur kurz auf dieses Verfahren für $d = 1$ eingegangen werden.

Man versucht β_0 und β_1 so zu wählen, dass die vorhergesagten Wahrscheinlichkeiten an den Stellen x_i der Trainingspunkte, so gut wie möglich mit der Klasse der jeweiligen Beobachtung zusammenpassen. Somit sollte die vorhergesagte Wahrscheinlichkeit für eine Beobachtung der Klasse 1 nahe beim Wert 1 liegen und im umgekehrten Fall nahe beim Wert 0. Dies kann mit Hilfe der Likelihood-Funktion

$$l(\beta_0, \beta_1) = \prod_{i:\, y_i = 1} p(x_i) \prod_{i':\, y_{i'} = 0} (1 - p(x_{i'}))$$

ausgedrückt werden. Dabei sei $p(x) = \mathbf{P}[Y = 1|X = x]$. Die Koeffizienten $\hat{\beta}_0$ und $\hat{\beta}_1$ werden nun so gewählt, dass die Likelihood-Funktion l für diese Parameter ein Maximum annimmt, vgl. hierzu [Jam+16, S. 133]. Für die algorithmische Vorgehensweise sei auf [HTF09, S. 120 ff.] Kapitel 4.4.1 verwiesen.

Logistische Regression kann auch bei mehr als zwei Klassen angewendet werden (vgl. [HTF09, S. 119 ff.]). Jedoch wird sie dafür in der Praxis selten angewendet, da die Diskriminanzanalyse für Klassifikation mit mehreren Kategorien laut [Jam+16, S. 137 f.] beliebter ist.

7.1.3 Lineare Diskriminanzanalyse (LDA)

Seien K Kategorien gegeben und bezeichne mit $\pi_k = \mathbf{P}[Y = k]$ die Wahrscheinlichkeit dafür, dass eine zufällige Beobachtung zur k-ten Kategorie gehört. Ferner sei $f_k(x)$ die bedingte Dichte von X zur Klasse $Y = k$. Mit Hilfe des Satzes von Bayes erhält man dann

$$\mathbf{P}[Y = k|X = x] = \frac{f_k(x)\pi_k}{\sum_{l=1}^{K} f_l(x)\pi_l}. \tag{7.5}$$

Die Idee der Diskriminanzanalyse ist es nun die Dichten f_k zu modellieren, da man aus (7.5) und der Kenntnis der π_k die für optimale Klassifikation notwendigen bedingten Wahrscheinlichkeiten $p_k(x) = \mathbf{P}[Y = k|X = x]$ berechnen kann.

Je nach Wahl der Dichten ergeben sich viele verschiedene Verfahren. Dazu gehören viele Klassifikatoren mit nichtlinearen Klassengrenzen, einer davon ist der naive Bayes-Klassifikator. Im Fall von Gauß-Funktionen erhält man die lineare und quadratische Diskriminanzanalyse.

Für die Wahl von f_k als multivariate Gauß-Funktion ist

$$f_k(x) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \cdot \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)\right). \tag{7.6}$$

Dabei ist Σ_k die Kovarianzmatrix der k-ten Klasse und μ_k der klassenspezifische Erwartungswert. Für Klassen k und l schaut man sich nun das logarithmische Verhältnis der bedingten Wahrscheinlichkeiten für die jeweilige Klasse an:

$$\ln\left(\frac{p_k(x)}{p_l(x)}\right) = \ln\left(\frac{\mathbf{P}[Y = k|X = x]}{\mathbf{P}[Y = l|X = x]}\right) = \ln\left(\frac{f_k(x)}{f_l(x)}\right) + \ln\left(\frac{\pi_k}{\pi_l}\right). \tag{7.7}$$

Die lineare Diskriminanzanalyse entsteht unter der Annahme, dass die Klassen eine gemeinsame Kovarianzmatrix $\Sigma_k = \Sigma$ für alle k haben. So wird aus (7.7) dann

$$\ln\left(\frac{p_k(x)}{p_l(x)}\right) = \ln\left(\frac{\pi_k}{\pi_l}\right) - \frac{1}{2}\left(\mu_k + \mu_l\right)^T \Sigma^{-1}(\mu_k - \mu_l) + x^T \Sigma^{-1}(\mu_k - \mu_l). \tag{7.8}$$

Das ist eine Gleichung linear in x. Damit ergibt sich analog zur logistischen Regression wie bei (7.4) eine Hyperebene als Klassengrenze. Diese Betrachtung kann man für alle Paare von Klassen k und l durchführen und deshalb sind alle Klassengrenzen linear.

Aus (7.8) leitet sich dann die lineare Diskriminanzfunktion

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \ln \pi_k$$

ab. Diese wird als äquivalente Beschreibung einer Entscheidungsregel benutzt, mit

$$Y(x) = \underset{k}{\operatorname{argmax}}\, \delta_k(x). \tag{7.9}$$

Da im Praxisfall die Parameter der Gauß-Verteilung nicht bekannt sind, müssen diese Geschätzt werden:

$$\begin{aligned}
\hat{\pi}_k &= \frac{n_k}{n}, \\
\hat{\mu}_k &= \sum_{y_i = k} \frac{x_i}{n_k}, \\
\hat{\Sigma} &= \frac{1}{n - K} \sum_{k=1}^{K} \sum_{y_i = k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T.
\end{aligned}$$

Dabei ist n_k die Anzahl der Beobachtungen, die zur Klasse k gehören, siehe [HTF09, S. 106 ff.]. In Abbildung 7 sieht man die Unterteilung der Klassen des in R vorhandenen Iris Datensatzes anhand der hier beschriebenen linearen Diskriminanzanalyse.

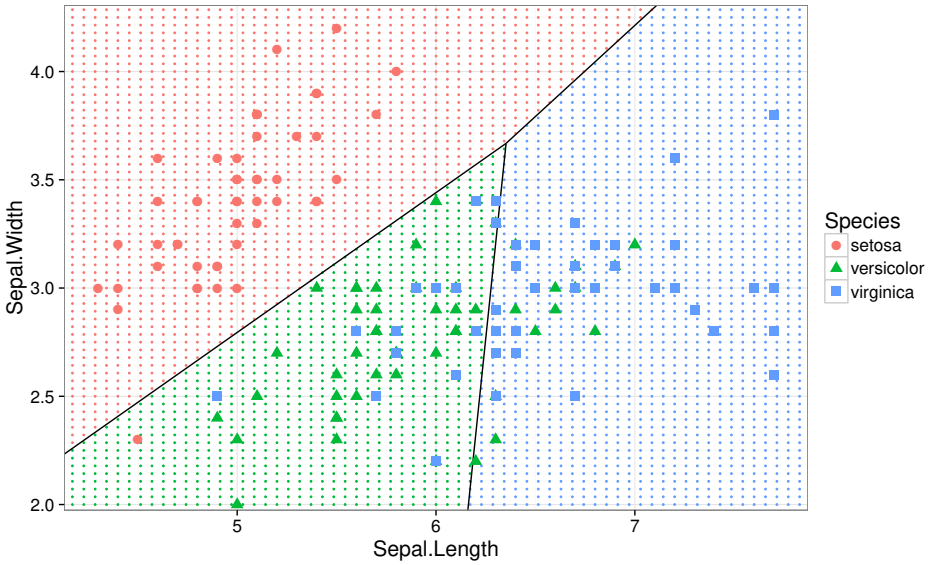


Abbildung 7: LDA angewendet auf den Iris Datensatz

Vergleicht man nun die logistische Regression mit der linearen Diskriminanzanalyse, so haben beide lineare Klassengrenzen. Allerdings sind laut [Jam+16, S. 138] für Klassen, die gut getrennt sind, die Parameterschätzungen für das logistische Regressionsmodell überraschenderweise instabil. Hingegen tritt dieses Problem bei der linearen Diskriminanzanalyse nicht auf. Ist n klein und die Verteilung der Prädiktoren X näherungsweise normal für jede Klasse, so ist die lineare Diskriminanzanalyse wieder stabiler. Für diesen Vergleich sei auch auf [HTF09, S. 127 ff.] verwiesen.

In [Jam+16, S. 151 ff.] wird dazu auch ein Vergleich zwischen logistischer Regression, k-Nächste-Nachbarn-Klassifikation, linearer und quadratischer Diskriminanzanalyse vorgenommen.

7.1.4 Quadratische Diskriminanzanalyse (QDA)

Die Quadratische Diskriminanzanalyse ist ein allgemeinerer Fall als die lineare Diskriminanzanalyse. Hier wird nicht die Bedingung gestellt, dass die Kovarianzmatrizen Σ_k in (7.6) gleich sind. Somit bleiben die quadratischen Terme erhalten und es lässt sich die folgende quadratische Diskriminanzfunktion aufstellen:

$$\delta_k(x) = -\frac{1}{2}\ln|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \ln\pi_k.$$

Damit sind auch die jeweiligen Klassengrenzen quadratisch. Wie bei Diskriminanzanalysen üblich, erhält man dann wieder den Klassifizierer (7.9), der als Klasse diejenige ausgibt, bei welcher die Diskriminanzfunktion den größten Wert annimmt, vgl. [HTF09, S. 110].

7.1.5 Support Vector Machines (SVM)

Die Support Vector Machine (SVM) ist die Verallgemeinerung eines einfachen und intuitiven Klassifizierers, dem sogenannten Maximal Margin Klassifikator. Dieser kann für binäre Klassifikation benutzt werden. Jedoch ist er in vielen Fällen nicht einsetzbar, da er als Voraussetzung benötigt, dass die zwei Klassen durch eine lineare Klassengrenze aufgeteilt werden können. Dabei hilft der Support Vector Klassifizierer (SVC), der eine Verallgemeinerung davon bildet und dieses Problem löst. Auch hier erhält man eine lineare Klassengrenze. Support Vector Machines bauen wiederum auf diesem Ansatz auf und liefern sogar nichtlineare Grenzen.

Um Verwirrungen zu vermeiden, sollte man diese drei Verfahren nicht unter dem Begriff Support Vector Machines zusammenfassen, sondern auf eine klare Unterscheidung achten, siehe hierzu [Jam+16, S. 337 ff.].

Hier wird im weiteren nur auf binäre Klassifikation eingegangen.

Können die Klassen durch eine Hyperebene getrennt werden, so existiert nicht nur eine, sondern unendlich viele solcher Hyperebenen. Denn eine Hyperebene kann leicht ein wenig rotiert oder verschoben werden, ohne mit den Beobachtungspunkten in Berührung zu kommen.

Ein Klassifizierer muss nun eine Hyperebene auswählen. Dabei ist die Hyperebene mit maximalem Rand eine natürliche Wahl. Sie wird auch optimal trennende Hyperebene genannt. Das ist die Hyperebene, die am weitesten von den Beobachtungspunkten entfernt liegt und man sich somit recht sicher sein kann, dass die zugeordnete Klasse stimmt. Für jeden solchen Punkt lässt sich der rechtwinklige Abstand zu dieser Ebene bestimmen. Der kleinste mögliche solche Abstand zur Ebene kennzeichnet den Rand. Anders gesagt, sucht man die Ebene, die die größte Minimaldistanz zu den Beobachtungspunkten besitzt. Seien $\beta_0, \beta = (\beta_1, \ldots, \beta_d)$ die Koeffizienten der Hyperebene mit maximalem Rand, so ordnet der Maximal Margin Klassifikator anhand des Vorzeichens von

$$f(x) = \beta_0 + \beta^T x$$

eine Klasse zu. Die Beobachtungspunkte, die genau auf dem Rand der Hyperebene liegen, werden dabei Stützvektoren genannt. Diese besitzen die Eigenschaft, dass bei ihrer Verschiebung sich auch die Hyperebene verschiebt. Letzten Endes hängt die Hyperebene nur von den Stützvektoren in dem Sinne ab, dass Verschiebung anderer Punkte, solange sie danach nicht zwischen Rand und Hyperebene liegen, keinen Einfluss auf die Ebene nehmen. Das kann man gut in Abbildung 8 erkennen.

Will man einen solchen Klassifizierer konstruieren, so hat man das folgende Optimierungsproblem zu lösen:

$$\begin{aligned} &\underset{\beta_0,\ldots,\beta_d}{\text{maximiere}}\, M, \\ &\text{mit der Bedingung } \sum_{j=1}^{d} \beta_j^2 = 1, \\ &y_i(\beta_0 + \beta^T x_i) \geq M \qquad \forall i \in \{1, \ldots, n\}. \end{aligned}$$

M repräsentiert die Spanne der Hyperebene.

Hat man nun den Fall, dass die zwei Klassen nicht durch eine Hyperebene trennbar sind, so kann man den Maximal Margin Klassifizierer nicht anwenden. Die Idee des Support Vector Klassifikators besteht nun darin, die Klassen nicht strikt zu trennen, sondern nur fast. Dazu werden sogenannte weiche Ränder benutzt. Deswegen nennt man den Klassifikator manchmal auch Soft Margin Klassifikator. Die weichen

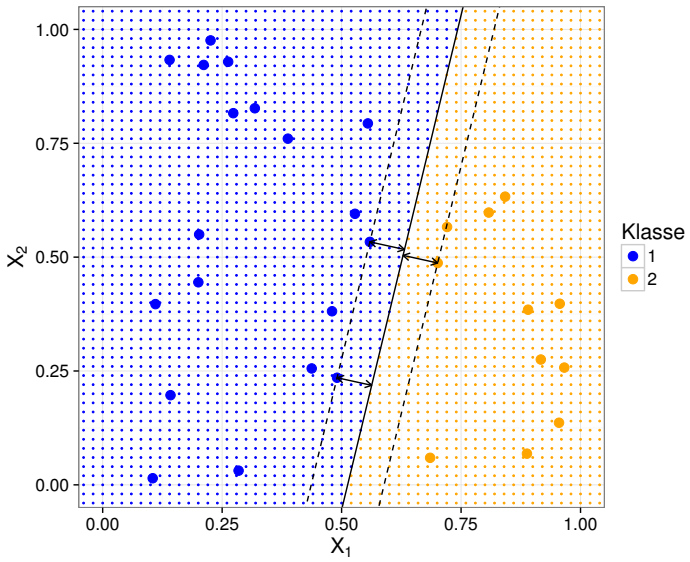


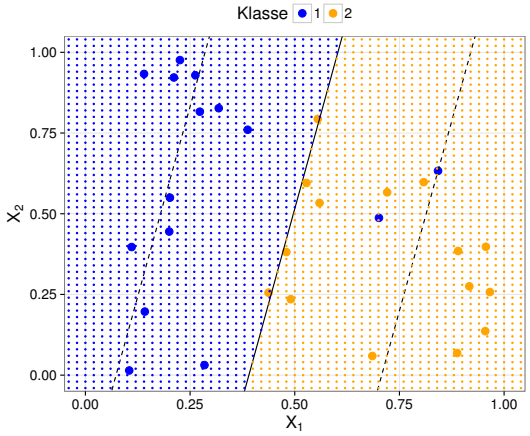
Abbildung 8: Beispiel für die Klassifizierung durch einen Maximal Margin Klassifizierer. Die durchgezogene Linie bildet dabei die vom Klassifikator erzeugte Hyperebene und der Abstand dieser Ebene zu den jeweils gestrichelten Geraden ist die maximale Spanne. Das wird für die drei Stützvektoren mittels Pfeilen angedeutet.

Ränder lassen auch falsche Klassifikationen zu. Dieser Klassifizierer wird auch im Fall separabler Klassen eingesetzt, um Überanpassung an die Beobachtungsdaten zu vermeiden oder auch um den Rand zu vergrößern und damit das Vertrauen in die Vorhersage. Denn die Entfernung des Punkte von der Hyperebene kann man als Maß für unsere Überzeugung interpretieren, dass eine richtige Klassifizierung vorliegt.

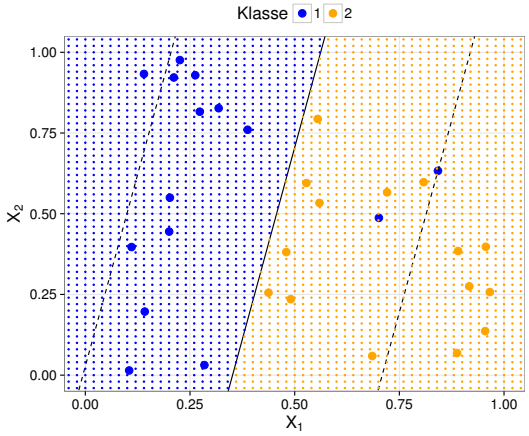
Das dadurch entstehende Optimierungsproblem sieht wie folgt aus:

$$\begin{aligned} &\underset{\beta_0,\ldots,\beta_d,\epsilon_1,\ldots\epsilon_d}{\text{maximiere}}\ M, \\ &\text{mit der Bedingung } \sum_{j=1}^{d} \beta_j^2 = 1, \\ &y_i(\beta_0 + \beta^T x_i) \geq M(1-\epsilon_i) \qquad \forall i \in \{1,\ldots,n\}, \\ &\epsilon_i \geq 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C. \end{aligned} \tag{7.10}$$

Hierbei ist C ein nichtnegativer Anpassungsparameter und M wieder die Breite des Randbereiches. Die $\epsilon_1, \ldots, \epsilon_d$ sind Schlupfvariablen, die den einzelnen Beobachtungspunkten erlauben, im Rand auf der falschen Seite der Hyperebene zu sein. Die ϵ_i geben dabei die Position der i-ten Beobachtung relativ zur Hyperebene und zum Rand gesehen an. Für $\epsilon_i = 0$ ist die Beobachtung auf der richtigen Seite außerhalb



(a) SVC mit $Cost = 2$ bei Verwendung des R Paketes e1071.



(b) SVC mit $Cost = 6$ bei Verwendung des R Paketes e1071.

Abbildung 9: Support Vector Klassifizierer.

des Randes. Für $\epsilon_i > 0$ ist sie auf der falschen Seite des Randes, was auch Übertreten des Randes genannt wird und für $\epsilon_i > 1$ ist die Beobachtung sogar auf der falschen Seite der Hyperebene. Der Parameter C spielt die Rolle eines Vorrats. Ist $C = 0$, so sind alle $\epsilon_i = 0$ und man erhält den Maximal Margin Klassifikator. Für $C > 0$ können maximal C Punkte auf der falschen Seite der Hyperebene liegen, da für diese $\epsilon_i > 1$ und $\sum_{i=1}^{n} \epsilon_i \leq C$ wegen (7.10) gilt. Vergrößert man also den Vorrat C, so gibt es mehr Punkte die den Rand übertreten dürfen. Gewöhnlicherweise wird dann der Rand breiter sein. Verkleinert man den Vorrat C, so werden weniger Übertretungen des Randes zugelassen und somit wird der Rand fast immer dünner, vgl. dazu Abbildung 9.

Ähnlich zum Maximal Margin Klassifizierer beeinflussen hier nur die Punkte, die auf dem Rand liegen oder ihn übertreten, die Lage der Hyperebene und damit den Klassifikator. Diese Punkte nennt man auch hier wieder Stützvektoren.

Um nun auch nichtlineare Klassengrenzen zu erzeugen, könnte man den Merkmalsraum vergrößern und zum Beispiel quadratische Terme der Prädiktoren hinzufügen. Damit würde man im erweiterten Merkmalsraum den Support Vector Klassifikator anwenden, jedoch im Orginalraum einen quadratischen Klassifizierer. Das vergrößert allerdings die Anzahl der Merkmale stark und führt häufig zu Dimensionalitätsproblemen. Dieses Problem wird bei Support Vector Machines elegant umgangen.

Support Vector Machines stellen eine Erweiterung des Support Vector Klassifizierers dar. Dabei wird der Merkmalsraum durch Benutzung von Kernen vergrößert.

Es lässt sich zeigen, dass der Support Vector Klassifizierer durch

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle , \tag{7.11}$$

mit n Parametern α_i repräsentiert werden kann. Außerdem stellt sich heraus, dass die α_i nur für Stützvektoren der Lösung ungleich 0 sind. Deshalb kann man (7.11) in der Form

$$f(x) = \beta_0 + \sum_{i \in I} \alpha_i \langle x, x_i \rangle \tag{7.12}$$

schreiben. Dabei sei I die Menge der Indizes der Stützpunkte. Die Idee besteht nun darin, das innere Produkt zu verallgemeinern. Dieses verallgemeinerte innere Produkt bezeichnen wir dann als Kern K. Der Kern misst die Ähnlichkeit von zwei Beobachtungen. Man erhält dann aus (7.12) als Klassifizierer

$$f(x) = \beta_0 + \sum_{i \in I} \alpha_i K(x, x_i).$$

Hier einige typische Kerne aus [HTF09, S. 424] entnommen:

$$\begin{aligned} \text{Polynom vom Grade } m\colon\ & K(x, x') = (1 + \langle x, x' \rangle)^m, \\ \text{Radial Basis}\colon\ & K(x, x') = \exp\left(-\gamma \left\| x - x' \right\|^2\right), \\ \text{Neuronales Netz}\colon\ & K(x, x') = \tanh(\kappa_1, \langle x, x' \rangle + \kappa_2). \end{aligned}$$

7.1.6 Entscheidungsbäume (Tree)

Entscheidungsbäume können sowohl zur Regression als auch zur Klassifikation benutzt werden. Sie beinhalten die Unterteilung des Merkmalsraum in eine gewisse Anzahl von einfachen Bereichen. Die Konstruktion eines Entscheidungsbaumes besteht im Allgemeinen aus zwei Schritten:

1. Man unterteilt den Merkmalsraum $\mathbf{X}$ in J verschiedene, sich nicht überlappende Regionen $R_1, \ldots, R_J$. Sie bilden eine Partition des Merkmalsraums $\mathbf{X}$.

2. Für jede zu klassifizierende Beobachtung bestimme man den zugehörigen Bereich R_j und weist die Klasse zu, die bei den Beobachtungsdaten dieses Bereichs am häufigsten vorkam.

Prinzipiell können die Regionen jede Form annehmen. Jedoch wird in den meisten Fällen mit d-dimensionalen Quadern gearbeitet. Dadurch bleibt das Modell einfach und ist leicht zu interpretieren. Da es nicht realisierbar ist, jede mögliche Partition

des Merkmalsraum in J Quader in Betracht zu ziehen, wählt man einen habgierigen Top-down-Ansatz, der rekursive Binärteilung genannt wird. Das heißt, dass der Merkmalsraum langsam immer weiter verfeinert wird. Also beginnend mit der Wurzel (top) bis zu den Blättern (down). Diese Strategie ist habgierig, weil bei jeder Verfeinerung des Raumes die beste in diesem Schritt mögliche Aufteilung vorgenommen wird. Es kann aber sein, dass es eine andere Unterteilung gibt, die sich erst später als optimal herausstellt und somit zu einem besseren Baum geführt hätte. Um die Teilung zu bewerkstelligen, muss man nun einen Prädiktor X_j und seine zugehörige Schnittstelle auswählen. Damit erhält man zwei Mengen $\{x \mid x^{(j)} < s\}$ und $\{x \mid x^{(j)} \geq s\}$, was eine Verfeinerung des Raumes darstellt. Daher kommt auch der Name Binärteilung, siehe [Jam+16, S. 303 ff.].

Es gibt verschiedene Kriterien nach denen man diese Auswahl treffen kann. Die gebräuchlichsten sind die Klassifikationsfehlerrate, der Gini-Index und die Kreuzentropie.

Man bezeichne nun mit $\hat{p}_{mk}$ den Anteil der Beobachtungsdaten in der m-ten Region der Klasse k. Setzt man die Klassifikationsfehlerrate E ein, so versucht man j und s so zu wählen, dass

$$E = 1 - \max_k \hat{p}_{mk}$$

minimiert wird. Jedoch ist das für die Baumkonstruktion nicht hinreichend empfindlich.

Dafür gibt es den Gini-Index

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

und die Kreuzentropie

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \ln(\hat{p}_{mk}).$$

Auch hier versucht man die Werte für G und D zu minimieren. Im Gegensatz zur Klassifikationsfehlerrate, sind der Gini-Index und die Kreuzentropie feinfühliger für die Reinheit der Knoten des Baumes. Das bedeutet, sie geben besser wieder, ob eine Klasse eine vorherrschende Rolle besitzt, siehe [Jam+16, S. 311 f.].

Zur Verbesserung des Testfehlers wird der Baum noch zurechtgestutzt, siehe dazu [Jam+16, S. 307 ff.].

7.1.7 Random Forest (RF)

Die Idee hinter Random Forest ist eine große Anzahl B an verschiedenen Entscheidungsbäumen anhand der Beobachtungsdaten zu konstruieren und dann eine Mehrheitsentscheidung der Bäume durchzuführen, vgl. [Jam+16, S. 319 ff.].

Für jeden Baum T_b nimmt man sich eine Stichprobe Z_b im Bootstrap-Verfahren (siehe [HTF09, S. 249]). Dazu werden zufällige Beobachtungen der Trainingsdaten mit der gleichen Anzahl wie die Trainingsdaten gewählt. Beobachtungen können auch mehrfach oder gar nicht auftreten. Zur Konstruktion des Baumes T_b führt man nun folgende Schritte mit der Stichprobe Z_b rekursiv für jedes Blatt des Baumes durch, bis man die minimale Knotengröße n_{min} erreicht hat:

1. Wähle m Variablen zufällig aus den d Prädiktoren aus.
2. Finde nun die beste Variable und Schnittstelle.
3. Unterteile den Knoten in 2 Unterknoten.

Für die Regression nimmt man nun die Funktion

$$f_B(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x).$$

Zur Klassifikation benutzt man die Mehrheitsentscheidung der Bäume T_b.

Dabei sind m und n_{min} als Tuningparameter anzusehen, siehe [HTF09, S. 588].

7.1.8 k-Nächste-Nachbarn-Klassifikation (KNN)

Bei diesem Verfahren verwendet man die Beobachtungen der Trainingsdaten, die im Eingaberaum $\mathbf{X}$ dem Punkt x am nächsten liegen, um eine Vorhersage $\hat{Y}$ zu erhalten. Der Klassifikator hat dann bei Einbeziehung von k Nachbarn die Form

$$\hat{Y}(x) = \sum_{x_i \in N_k(x)} \omega_i y_i,$$

wobei typischerweise $\omega_i = \frac{1}{k}$ gesetzt wird und $N_k(x)$ die Nachbarschaft von x definiert, welche aus den k nächsten Punkten x_j der Trainingsdaten besteht. Die Nähe wird durch eine Metrik gemessen. Gewöhnlicherweise wählt man den euklidischen Abstand. Man beachte, dass mehrere Punkte die gleiche Entfernung besitzen können. Ist das der Fall, so wird zum Beispiel der Punkt bevorzugt, der bei Nummerierung der Trainingsdaten den kleineren Index hat, siehe dazu [DGL96, S. 61 f.].

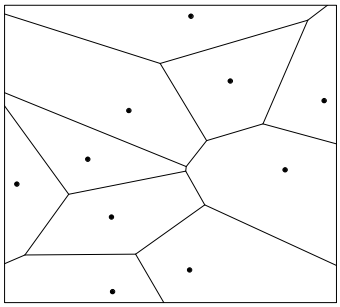


Abbildung 10: Voronoi-Diagramm.

Zur Einteilung in zwei Klassen benutzt man dann wieder einen Schwellwert. Für $\mathbf{Y} = \{-1, 1\}$ und $\omega_i = \frac{1}{k}$ für alle i, kann dann folgender Klassifikator gewählt werden:

$$f(x) = \begin{cases} 1, & \text{falls } \hat{Y}(x) > 0, \\ -1, & \text{sonst.} \end{cases}$$

Für $k = 1$ können die Klassifikationsbereiche recht einfach berechnet werden und entsprechen bei euklidischer Metrik dem Voronoi-Diagramm. Jeder Trainingspunkt x_i besitzt einen Bereich, für den er der nächste Nachbar ist. vgl. [HTF09, S. 14 f.]. Wie eine solche Unterteilung für $d = 2$ aussehen kann, sieht man in Abbildung 10.

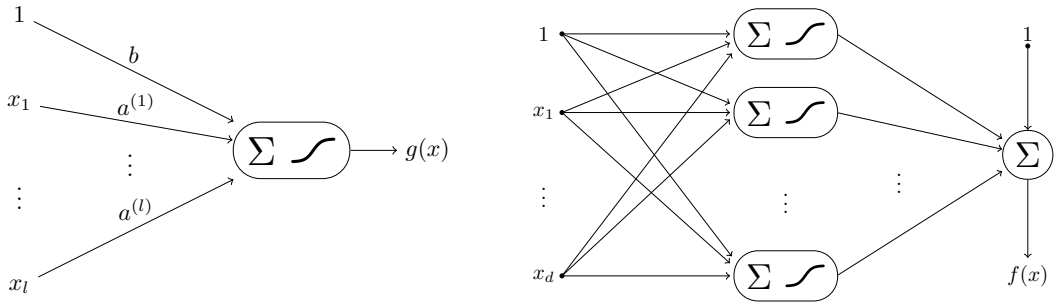
7.1.9 Neuronale Netze (NNet)

Künstliche neuronale Netze sind durch den Wunsch entstanden, das menschliche Gehirn mit Hilfe eines Computers zu modellieren. Sie bestehen aus vielen Recheneinheiten, den Neuronen. Die Neuronen erhalten eine gewisse Anzahl an Eingangssignalen und errechnen daraus ein Ausgabesignal. In der Orginalfassung produzierten die künstlichen Neuronen nur binäre Ausgaben in Anlehnung an die biologischen Verwandten, die nur feuern, wenn ein Schwellwert durch die Eingabesignale überschritten wird. Die Neuronen können ebenfalls wie beim Gehirn miteinander verbunden werden. Jedoch enden hier die Gemeinsamkeiten mit den biologischen neuronalen Netzen, denn die künstlichen neuronalen Netze stellen nur ein sehr vereinfachtes Modell dar. Die Anzahl der Neuronen und Verbindungen ist stark beschränkt.

Ein künstliches Neuron wird durch eine reellwertige Funktion des $\mathbb{R}^l$

$$g(x) = \sigma(a^T x + b)$$

beschrieben. Dabei ist $x \in \mathbb{R}^l$ der Eingabevektor und $a^T = (a^{(1)}, \ldots, a^{(l)}) \in \mathbb{R}^l, b \in \mathbb{R}$ sind Gewichte. $\sigma\colon \mathbb{R} \to [0;1]$ wird Sigmoidfunktion genannt, siehe hierzu [Gyö+02, S. 297 ff.]. Abbildung 11a zeigt eine Darstellung eines künstlichen Neurons.



(a) Ein künstliches Neuron. (b) Ein künstliches neuronales Netz.

Abbildung 11: Aufbau neuronaler Netze.

Es gibt viele verschiedene Arten von neuronalen Netzen. Häufig wird das sogenannte aufgeschaltete neuronale Netzwerk benutzt. In Abbildung 11b wird ein solches Netzwerk dargestellt. Es gibt keine Ausgabeverbindungen der Neuronen in eine vorhergehende Schicht. Hier wird es mit einer verdeckten Schicht von k Neuronen als reellwertige Funktion des $\mathbb{R}^d$ in der Form

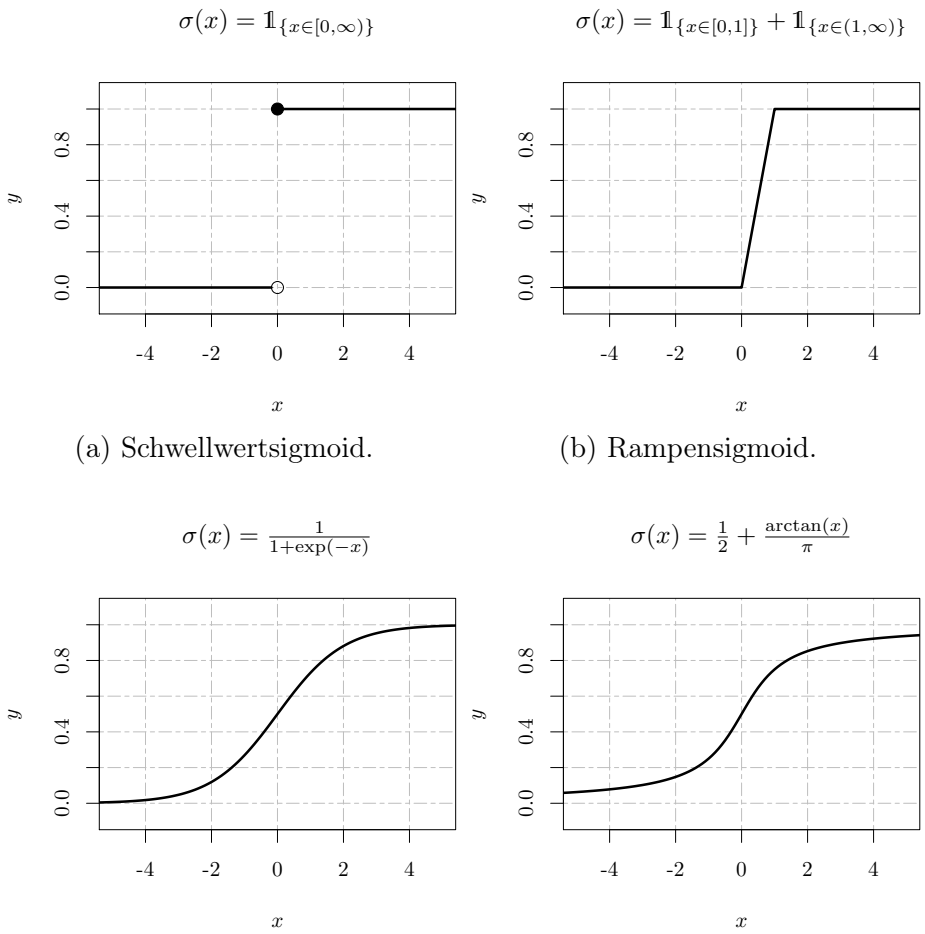
$$f(x) = \sum_{i=1}^{k} c_i \sigma(a_i^T x + b_i) + c_0$$

modelliert. Dabei seien $a_1, \ldots, a_k \in \mathbb{R}^d, b_1, \ldots, b_k, c_0, \ldots, c_k \in \mathbb{R}$ die Parameter dieses Modells und σ sei wieder eine Sigmoidfunktion. Typischerweise wird dabei eine gequetschte Sigmoidfunktion benutzt, das heißt σ ist nicht fallend und es gilt

$$\lim_{x \to -\infty} \sigma(x) = 0 \qquad \text{und} \qquad \lim_{x \to \infty} \sigma(x) = 1.$$

Beispiele für solche Funktionen sind das Schwellwertsigmoid Abbildung 12a, das Rampensigmoid Abbildung 12b, das logistische Sigmoid Abbildung 12c und das Arcustangenssigmoid Abbildung 12d. Der Graph nimmt dabei oft eine S-Form an.

Die Wahl der Parameter des neuronalen Netzes erfolgt für Regression über die Summe der quadratischen Fehler und für Klassifikation wird auch der quadratische Fehler oder die Kreuzentropie benutzt. Der allgemeine Ansatz ist die zugehörige Fehlerfunktion zu minimieren. Das erfolgt durch Gradientenabstiegsverfahren und wird für neuronale Netze Rückwärtspropagierung genannt, vgl. [HTF09, S. 395 ff.].



(a) Schwellwertsigmoid.

(b) Rampensigmoid.

(c) logistisches Sigmoid.

(d) Arcustangenssigmoid.

Abbildung 12: Sigmoidfunktionen.

7.1.10 Kreuzvalidierung

Auch wenn Kreuzvalidierung kein Klassifikationsverfahren ist, soll es hier kurz beschrieben werden, da es häufig zur Bewertung dieser eingesetzt wird.

Angenommen man besitzt einen Datensatz und will für diesen herausfinden, welches Verfahren die besten Prognosen liefert. Ein guter Ansatz dafür ist die Beobachtungen in eine Trainingsmenge und eine Validierungsmenge zu unterteilen. Mit der Trainingsmenge passt man das jeweilige Modell an und benutzt die Validierungsmenge zur Schätzung des Testfehlers für das Modell. Jedoch kann abhängig von der Unterteilung in diese zwei Mengen, die Schätzung des Fehlers stark variieren, da man für unterschiedliche Unterteilungen auch unterschiedliche Modelle und Testbeobachtungen erhält. Des Weiteren wird nur ein Teil der Beobachtungen genutzt, um das vorläufige Modell zu trainieren. Das ist ein Problem, denn üblicherweise schneiden Modelle mit weniger Trainingsdaten schlechter ab. Das legt den Schluss nahe, dass die wirkliche Testfehlerrate überschätzt wird.

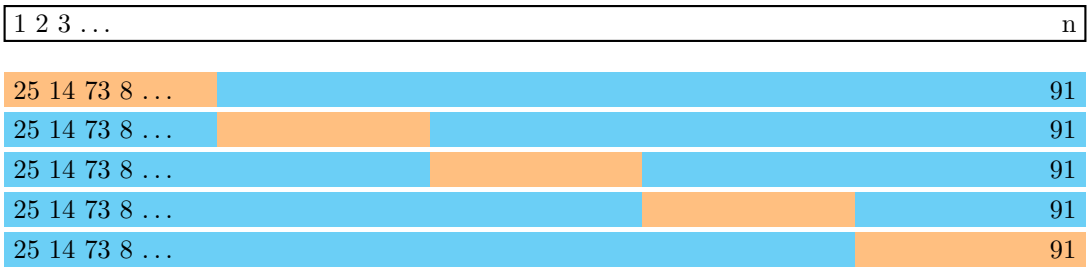


Abbildung 13: Eine schematische Darstellung einer 5-fachen Kreuzvalidierung mit n Beobachtungen und zufälliger Aufteilung in sich nicht überlappende Gruppen. Jede der fünf unteren Zeilen stellt eine Unterteilung der ursprünglichen Menge in Trainings- (blau) und Validierungsmenge (orange) dar.

Kreuzvalidierung wird genutzt, um diese zwei Probleme abzumildern. Dazu wird die Menge der Beobachtungen in k zufällige, ungefähr gleichgroße Gruppen eingeteilt. Danach wird eine der Gruppen als Validierungsmenge behandelt und die restlichen Gruppen bilden die Trainingsmenge. Damit schätzt man nun den Fehler. Wendet man das Verfahren so an, dass jede Gruppe einmal die Validierungsmenge war, so ergeben sich k Fehler. Bildet man das arithmetische Mittel dieser Fehler, so erhält man eine bessere Schätzung für den wirklichen Testfehler. Das beschriebene Verfahren nennt man dann auch k-fache Kreuzvalidierung. Abbildung 13 zeigt dieses Vorgehen als schematische Darstellung an einem Beispiel mit 5-facher Kreuzvalidierung.

Typische Werte in der Praxis sind meist $k = 5$ oder $k = 10$, da mit steigendem k auch die Anzahl der anzupassenden Modelle wächst und dieser Vorgang schnell sehr rechenintensiv werden kann, vgl.[Jam+16, S. 176 ff.].

7.1.11 Dimensionsreduzierung

In der Praxis kann ein Datensatz schnell eine große Anzahl an Merkmalen aufweisen, um möglichst alle vorhanden Informationen zu erfassen. Jedoch steigt damit in den meisten Fällen auch die Komplexität des zu trainierenden Modells und somit die benötigte Rechenzeit, sowie der Speicherbedarf. Um diesem Problem entgegenzusteuern, kann es hilfreich sein eine Dimensionsreduzierung vorzunehmen. Dabei gehen allerdings Informationen verloren. Deshalb sollte man diesen Verlust so gering wie möglich halten. Eine Möglichkeit stellt die Hauptkomponentenanalyse dar.

7.1.12 Hauptkomponentenanalyse

Bei der Hauptkomponentenanalyse versucht man eine gewisse Anzahl von Komponenten zu finden, die zusammengenommen den größten Teil der Variabilität in

den Daten erklärt. Die Hauptkompontenrichtungen sind dabei die Richtungen im Merkmalsraum, entlang denen sich die ursprünglichen Daten am meisten unterscheiden. Diese Komponenten sind eine Linearkombination der ursprünglichen Merkmale $X_1, \ldots, X_d$. Die i-te Hauptkomponente Z_i hat dann folgende Form:

$$Z_i = \sum_{j=1}^{d} \phi_{j,i} X_j,$$

wobei die Koeffizienten normalisiert sind, d.h. es gilt

$$\sum_{j=1}^{d} \phi_{j,i}^2 = 1.$$

Eine andere Interpretation der Hauptkomponenten ist, dass sie eine niedrigdimensionale lineare Mannigfaltigkeit bilden, die am nächsten zu den Beobachtungen liegt. Das bedeutet, dass die Hauptkomponenten die lineare Mannigfaltigkeit aufspannen, die die Summe der Quadrate des euklidischen Abstands von jedem Punkt zur Ebene minimieren, siehe [Jam+16, S. 374 ff.]. Die Hauptkomponenten erhält man häufig über die Singulärwertzerlegung (vgl. [HTF09, S. 535]), aber auch über Eigenwerte und zugehörige Eigenvektoren, siehe dazu [RUL14, S. 412 ff.].

7.2 Vergleich verschiedener Verfahren mit Hilfe der freien Software Umgebung R

Dieser Abschnitt beinhaltet den Vergleich der Verfahren des vorherigen Kapitels 7.1 mit den in Kapitel 6 selbst implementierten Verfahren CUDTP und ODT für ausgewählte Beispiele. Die Gegenüberstellung erfolgt anhand der Kennzahlen aus den Kontingenztafeln der Versuche, da zum Beispiel die Genauigkeit, also das Verhältnis von richtig klassifizierten Beobachtungen zu allen Beobachtungen, alleine nur unzureichend die Leistung des Klassifizierers wiedergibt. Wie die Kenngrößen aus den Kontingenztafeln gebildet werden, ist in Anhang B nachzulesen.

Zuerst werden drei künstlich generierte Datensätze gegenübergestellt. Sie stellen einen Kreis, ein Schachbrettmuster und eine Diagonale von links oben nach rechts unten dar. Dabei wurde kein Rauschen hinzugefügt. Zum Schluss werden die Verfahren mit Daten aus der Praxis eines Unternehmens ausgewertet.

Die Angaben zur verwendeten Hard- und Software können im Anhang C nachgelesen werden. Um reproduzierbare Ergebnisse zu gewährleisten, wurde jeweils für alle Funktionen mit Zufallszahlen ein gemeinsamer Startwert zur Generierung

der Zufallszahlen von 10 vorgegeben. Für die drei künstlichen Datensätze kamen für die jeweiligen Verfahren immer die gleichen Parameterwerte zum Einsatz. Für CUDTP sind das $nbar = 800$, $max\_subdivisions = 30$, bei ODT $kmax = 7$ und $gamma = 0{,}001$, sowie $k = 1$ für KNN und bei NNet $size = 3$ und $maxit = 300$. Ansonsten wurden Vorgabewerte genutzt.

7.2.1 Kreis

Für den Kreis werden jeweils 1000 Trainings- und Testbeobachtungen erstellt. Die zwei Prädiktorvariablen werden dabei jeweils zufällig in $[0; 1]$ verteilt und je nachdem, ob der entstandene Punkt im Kreis mit Mittelpunkt $(0{,}5; 0{,}5)$ und Radius von 0,4 Einheiten liegt, wird dem Zielmerkmal der Wert 2 oder außerhalb der Wert 1 zugewiesen. Erfasst werden die Kontingenztafeln (Tabelle 7.1) nach dem Training der einzelnen Verfahren auf den Testbeobachtungen, sowie sich daraus ergebende Kenngrößen (Tabelle 7.2) wie zum Beispiel die Klassifikationsgenauigkeit.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	519	25	517	18	511	32	196	145	196	145	541	66	536	23	521	24	521	21	356	79
2	22	434	24	441	30	427	345	314	345	314	0	393	5	436	20	435	20	438	185	380

Tabelle 7.1: Kontingenztafeln für einen Kreis bei Anwendung verschiedener Klassifikationsverfahren.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	0,953	0,958	0,938	0,510	0,510	0,934	0,972	0,956	0,959	0,736
Sensitivität	0,959	0,956	0,945	0,362	0,362	1,000	0,991	0,963	0,963	0,658
Spezifität	0,946	0,961	0,930	0,684	0,684	0,856	0,950	0,948	0,954	0,828
Positiver Vorhersagewert	0,954	0,966	0,941	0,575	0,575	0,891	0,959	0,956	0,961	0,818
Negativer Vorhersagewert	0,952	0,948	0,934	0,476	0,476	1,000	0,989	0,956	0,956	0,673
Prävalenz	0,541	0,541	0,541	0,541	0,541	0,541	0,541	0,541	0,541	0,541
Erkennungsrate	0,519	0,517	0,511	0,196	0,196	0,541	0,536	0,521	0,521	0,356
Erkannte Prävalenz	0,544	0,535	0,543	0,341	0,341	0,607	0,559	0,545	0,542	0,435
Ausbalancierte Genauigkeit	0,952	0,958	0,937	0,523	0,523	0,928	0,970	0,955	0,959	0,743

Tabelle 7.2: Kenngrößen zu den Kontingenztafeln des Kreises aus Tabelle 7.1 zur Beurteilung der Klassifikatoren.

Zur besseren Veranschaulichung der beiden Verfahren CUDTP und ODT wurden die Testdaten mit den jeweiligen Partitionierungsgrenzen und deren Klassifikationsfarben abgebildet. In Abbildung 14 ist das für CUDTP geschehen, jedoch wurden nicht die gesamten Trainingsdaten, sondern nur die ersten 800 gezeichnet. Das liegt daran, dass die restlichen 200 Trainingsdaten zur Auswahl der besten Partitionierung gebraucht werden und nicht direkt in die Klassifizierung der Bereiche eingehen. Somit lässt sich anhand des linken Bildes gut die Einfärbung der Bereiche des rechten

Bildes nachvollziehen. Beim rechten Bild fällt auf, dass am rechten Rand scheinbar ein Quadrat falsch eingefärbt ist. Schaut man jedoch ins linke Bild, so sieht man, dass in diesem Quadrat keine Beobachtung enthalten ist. Das Standardverhalten des Klassifikators ist dann, die Klasse, die am häufigsten in den Trainingsdaten vorkommt, einzusetzen. In unserem Fall ist das die Klasse 2 mit der Farbe Orange. Ein weiteres leeres Quadrat gibt es am oberen Rand des Kreises. Das fällt jedoch nur beim Zeichnen der Trainingsdaten auf, weil es genau an der Klassengrenze liegt.

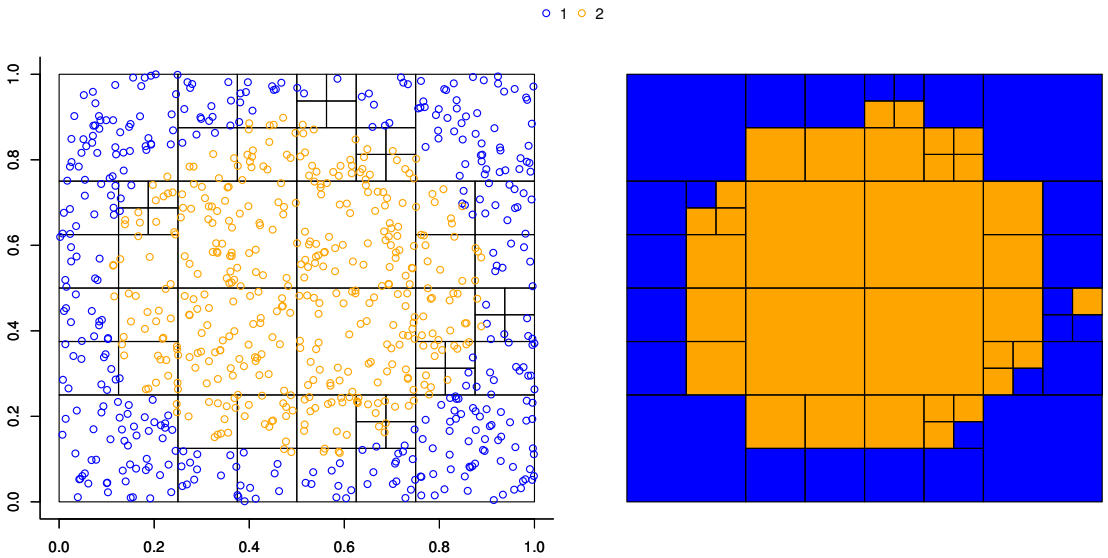


Abbildung 14: Trainingsdaten für die Klassifikation eines Kreises und die zugehörige Einteilung des Klassifikators mittels CUDTP.

Bei der Abbildung des Kreises (Abbildung 15) durch ODT gibt es keine leeren Bereiche. Zum einen löst ODT die beiden Bereiche mit den leeren Quadraten gröber auf und zum anderen sind dort auch mehr Punkte vorhanden.

Bei beiden Verfahren ist der Kreis bei dieser Anzahl an Trainingsdaten gut erkennbar und wie zu erwarten bei adaptiven Methoden, werden die Klassengrenzen fein aufgelöst.

Sieht man sich nun die Kenngrößen in Tabelle 7.2 an, so zeigt sich, dass die Verfahren CUDTP, ODT, Tree, QDA, SVM, RF und KNN sehr gut abschneiden. Das neuronale Netz ist bei der Parameterwahl von 3 verdeckten Knoten nicht ganz so gut. Weit abgeschlagen sind hingegen die linearen Klassifizierer Logit und LDA, da die Klassengrenzen stark von linearen Grenzen abweichen. Spitzenreiter ist in diesem Test das SVM Verfahren. Allerdings liegen andere Verfahren wie CUDTP und ODT nur wenige Prozentpunkte bzgl. der Genauigkeit dahinter. Es ist nicht auszuschließen, dass sich das Ranking bei verschiedener Parameterwahl verschieben würde. Speziell könnte das für CUDTP und ODT mit stärker verzweigten Bäumen möglich sein.

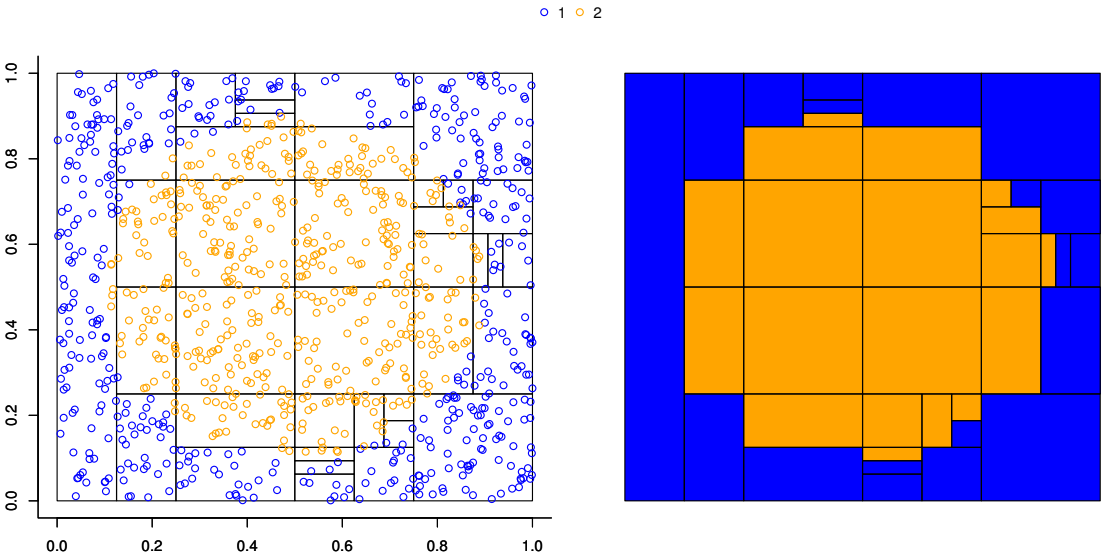


Abbildung 15: Trainingsdaten für die Klassifikation eines Kreises und die zugehörige Einteilung des Klassifikators mittels ODT.

7.2.2 Schachbrettmuster

Auch bei diesem Muster wurden wieder 1000 Trainings- und Testbeobachtungen generiert und die Prädiktorvariablen liegen in $[0;1]$. Es besteht aus 64 gleichgroßen Quadraten mit sich jeweils abwechselnden Klassen. Dem Quadrat beginnend bei $(0;0)$ wird dabei die Klasse 1 zugeordnet und erhält in den Abbildungen die Farbe Blau. Klasse 2 bekommt die Farbe Orange. Es stellt also ein Schachbrettmuster dar.

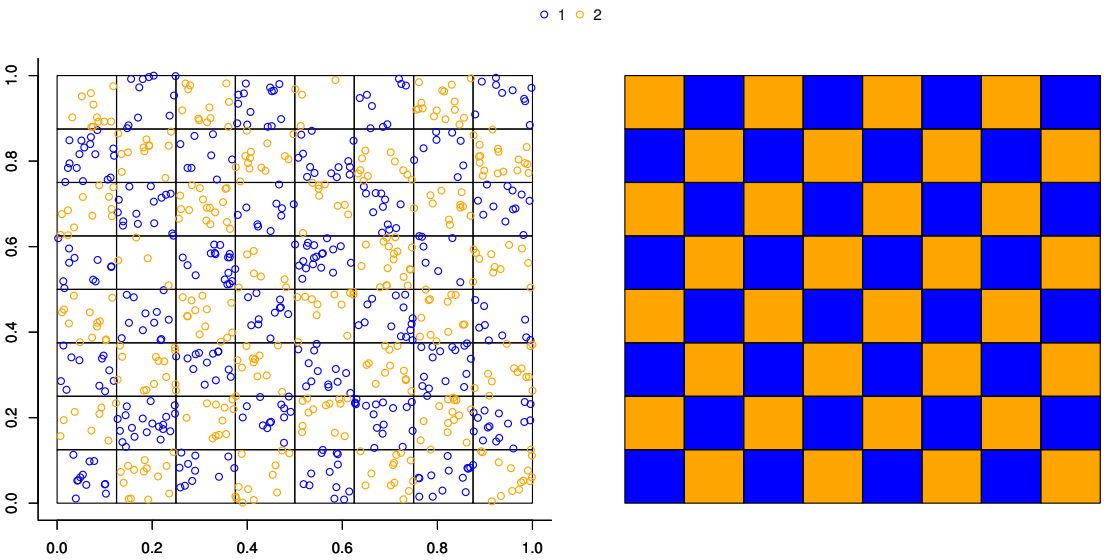


Abbildung 16: Trainingsdaten für die Klassifikation eines Schachbrettmusters und die zugehörige Einteilung des Klassifikators mittels CUDTP.

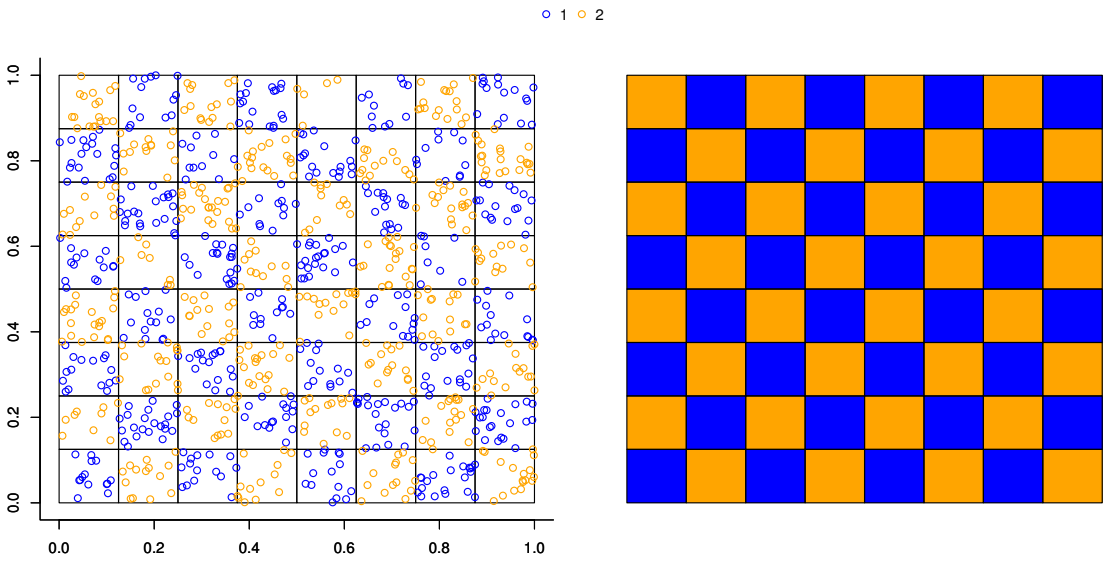


Abbildung 17: Trainingsdaten für die Klassifikation eines Schachbrettmusters und die zugehörige Einteilung des Klassifikators mittels ODT.

Betrachtet man wieder die Kenngrößen aus Tabelle 7.4, so fällt auf, dass CUDTP und ODT hier perfekte Werte erreichen und damit keine Fehler beim Klassifizieren begehen. Das Beispiel zeigt den Optimalfall für beide Verfahren. Zum einen bilden die Klassengrenzen Linien, die parallel zu einer der Achse liegen und zum anderen sind sie genau an den Stellen, an denen auch die Unterteilungen der Bereiche des Klassifikators erfolgen. Die linearen Verfahren Logit und LDA sind auch hier wieder nicht geeignet. Obwohl SVM beim Kreis ein annähernd perfektes Ergebnis mit 97 % ausgeglichener Genauigkeit hatte und auch Tree und QDA sehr gute Ergebnisse lieferten, haben sie hier wie auch NNet unzureichende Genauigkeit und sind kaum oder gar nicht besser als die linearen Verfahren. Hingegen beschreiben RF mit 83 % ausgeglichener Genauigkeit und KNN mit fast 87 % ausgeglichener Genauigkeit das Muster recht gut.

Die Entstehenden Partitionierungen sind für CUDTP und ODT in diesem Fall gleich und man sieht ein perfektes Schachbrettmuster auf der rechten Seite von Abbildung 16 und Abbildung 17.

7.2.3 Diagonale Fläche

Die Voraussetzungen dieses Anwendungsbeispiels sind wie beim Kreis und dem Schachbrettmuster. Es gibt zwei Bereiche der Klasse 1, die jeweils ein gleichschenkliges Dreieck mit rechtem Winkel und einer Schenkellänge von 0,6 Einheiten bilden. Die Spitze der rechten Winkel liegt in $(0; 0)$ und $(1; 1)$.

Auch hier (Tabelle 7.6) sind die linearen Verfahren Logit und LDA ungeeignet und liefern einen Klassifizierer der jeder Beobachtung die Klasse 2 zuweist. Alle anderen Verfahren schneiden hier sehr gut ab und haben ausbalancierte Genauigkeiten von weit über 90 %. ODT, Tree und QDA liegen hierbei ungefähr gleich auf. CUDTP hat eine leicht höhere Genauigkeit, die jedoch von SVM, RF, KNN und NNet noch um einige Prozente verbessert wird. In diesem Fall wird der beste Klassifizierer von NNet erstellt. Er begeht bei nur 1 aus 1 000 Testklassifikationen einen Fehler.

Abbildung 18 und Abbildung 19 geben die diagonale Fläche gut wieder und man erkennt die genauere Auflösung an den Klassengrenzen.

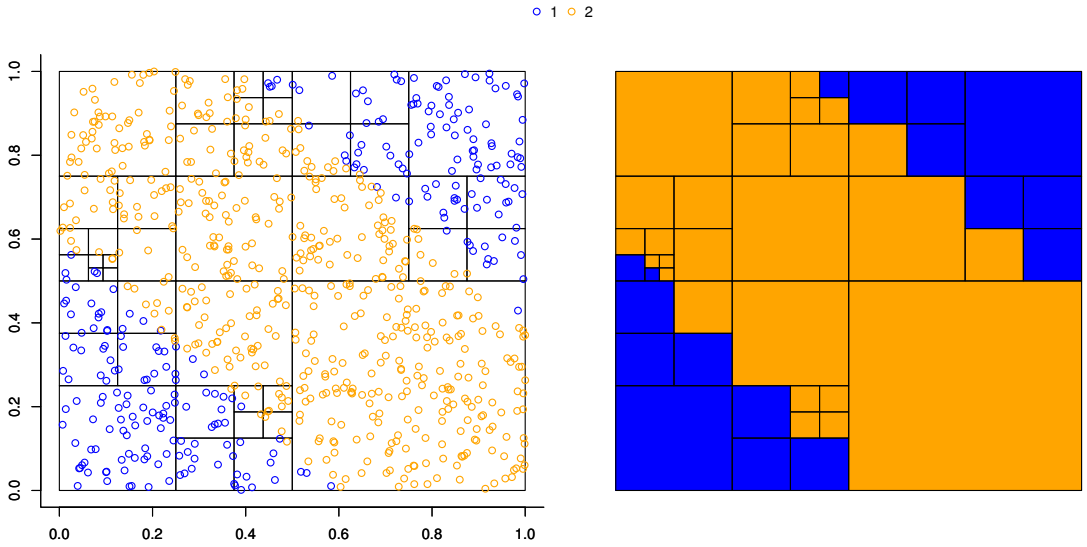


Abbildung 18: Trainingsdaten für die Klassifikation einer diagonal liegenden Fläche und die zugehörige Einteilung des Klassifikators mittels CUDTP.

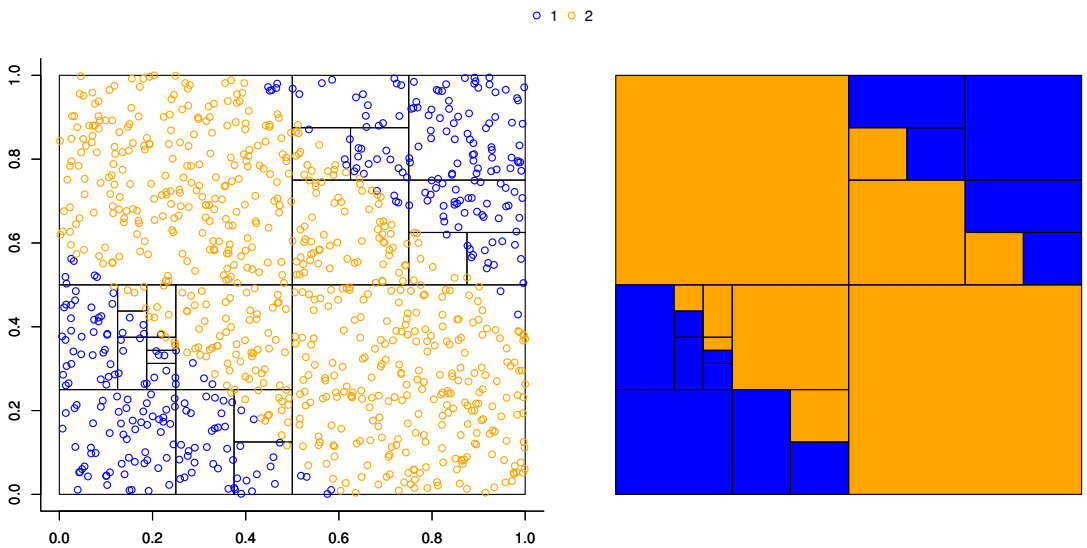


Abbildung 19: Trainingsdaten für die Klassifikation einer diagonal liegenden Fläche und die zugehörige Einteilung des Klassifikators mittels ODT.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	486	0	486	0	486	514	328	357	328	357	261	265	392	390	401	83	420	66	411	393
2	0	514	0	514	0	0	158	157	158	157	225	249	94	124	85	431	66	448	75	121

Tabelle 7.3: Kontingenztafeln für ein Schachbrettmuster bei Anwendung verschiedener Klassifikationsverfahren.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	1,000	1,000	0,486	0,485	0,485	0,510	0,516	0,832	0,868	0,532
Sensitivität	1,000	1,000	1,000	0,675	0,675	0,537	0,807	0,825	0,864	0,846
Spezifität	1,000	1,000	0,000	0,305	0,305	0,484	0,241	0,839	0,872	0,235
Positiver Vorhersagewert	1,000	1,000	0,486	0,479	0,479	0,496	0,501	0,829	0,864	0,511
Negativer Vorhersagewert	1,000	1,000	NaN	0,498	0,498	0,525	0,569	0,835	0,872	0,617
Prävalenz	0,486	0,486	0,486	0,486	0,486	0,486	0,486	0,486	0,486	0,486
Erkennungsrate	0,486	0,486	0,486	0,328	0,328	0,261	0,392	0,401	0,420	0,411
Erkannte Prävalenz	0,486	0,486	1,000	0,685	0,685	0,526	0,782	0,484	0,486	0,804
Ausbalancierte Genauigkeit	1,000	1,000	0,500	0,490	0,490	0,511	0,524	0,832	0,868	0,541

Tabelle 7.4: Kenngrößen zu den Kontingenztafeln eines Schachbrettmusters aus Tabelle 7.3 zur Beurteilung der Klassifikatoren.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	341	6	328	6	331	5	0	0	0	0	334	12	354	0	364	4	370	7	379	1
2	38	615	51	615	48	616	379	621	379	621	45	609	25	621	15	617	9	614	0	620

Tabelle 7.5: Kontingenztafeln für eine diagonal liegende Fläche bei Anwendung verschiedener Klassifikationsverfahren.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	0,956	0,943	0,947	0,621	0,621	0,943	0,975	0,981	0,984	0,999
Sensitivität	0,900	0,865	0,873	0,000	0,000	0,881	0,934	0,960	0,976	1,000
Spezifität	0,990	0,990	0,992	1,000	1,000	0,981	1,000	0,994	0,989	0,998
Positiver Vorhersagewert	0,983	0,982	0,985	NaN	NaN	0,965	1,000	0,989	0,981	0,997
Negativer Vorhersagewert	0,942	0,923	0,928	0,621	0,621	0,931	0,961	0,976	0,986	1,000
Prävalenz	0,379	0,379	0,379	0,379	0,379	0,379	0,379	0,379	0,379	0,379
Erkennungsrate	0,341	0,328	0,331	0,000	0,000	0,334	0,354	0,364	0,370	0,379
Erkannte Prävalenz	0,347	0,334	0,336	0,000	0,000	0,346	0,354	0,368	0,377	0,380
Ausbalancierte Genauigkeit	0,945	0,928	0,933	0,500	0,500	0,931	0,967	0,977	0,982	0,999

Tabelle 7.6: Kenngrößen zu den Kontingenztafeln der diagonal liegenden Fläche aus Tabelle 7.5 zur Beurteilung der Klassifikatoren.

7.2.4 Datensatz aus der Praxis

Die hier benutzten Daten wurden von der Firma Nugg.ad zur Verfügung gestellt. Nach importieren der CSV Datei in R, liegen die Informationen in Form eines data.frames mit 32 134 Zeilen und 310 Spalten vor. Um mit den Daten zu arbeiten, müssen sie noch in brauchbare Form gebracht werden. Dazu werden Spalten, die Klassennamen beinhalten, gesondert abgespeichert und solche, die Prädiktorvariablen darstellen, in einer Matrix zusammengefasst. In diesem Schritt werden auch Variablen entfernt, die keine Informationen enthalten, weil sie in jeder Zeile den gleichen Wert besitzen. Die entstehende Matrix hat 32 133 Zeilen und 300 Spalten, also 32 133 Beobachtungen mit jeweils 300 Merkmalen. Allerdings gibt es bei den Beobachtungen viele leere Merkmalswerte. Da viele Verfahren mit fehlenden Werten nicht umgehen können, muss man dafür Ersatzwerte einsetzen. Bei dieser Auswertung wurden Fehlwerte durch den Wert 0 substituiert, was zu Informationsverlust führen kann und somit die Endergebnisse beeinflusst. Die Merkmalswerte liegen schon im Intervall $[0; 1]$ und wären somit bereit zur weiteren Analyse und Klassifikation. Man beachte, dass die große Anzahl an Merkmalen für komplexe Klassifikationsmethoden zu Problemen wie langer Rechenzeit und hohem Speicherverbrauch führen kann. Das ist der Grund, weswegen wir die Dimension des Problems reduzieren müssen. Dazu wenden wir das in Abschnitt 7.1.12 beschriebene Verfahren der Hauptkomponentenanalyse an. Damit ergibt sich wieder eine Matrix mit 300 Spalten, allerdings beschreiben diese nun die Hauptkomponenten. Die erste Spalte stellt dann die erste Hauptkomponente dar. Jedoch liegen die Werte nicht mehr im Bereich $[0; 1]$ und müssen skaliert werden, weil die Verfahren CUDTP und ODT nur für solche Daten ausgelegt sind.

Will man nun beurteilen, wie erfolgreich die einzelnen Verfahren auf diesem Datensatz sind, so wird man das mit Hilfe von Kreuzvalidierung (Kapitel 7.1.10) bewerkstelligen. Denn man hat keinen zusätzlichen Testdatensatz und möchte in die Modellbildung so viele Beobachtungen wie möglich einbeziehen. Wir verwenden hier 5-fache Kreuzvalidierung, um die benötigte Rechenzeit zu verringern. Die Auswertung erfolgt mit aufsummierten Kontingenztafeln zur Abschätzung der Kenngrößen.

Es werden insgesamt drei Vergleiche durchgeführt. Die ersten beiden erfolgen mit fünf Hauptkomponten und unterscheiden sich hauptsächlich in der Parameterwahl für CUDTP und ODT. Der letzte Vergleich benutzt 13 Hauptkomponenten und spiegelt immer noch nur einen Bruchteil der zur Verfügung stehenden Informationen wieder. Mehr Hauptkomponenten führen dazu, dass für CUDTP und ODT zu viel Rechenzeit und später auch Speicherplatz benötigt wird. Um das zu verdeutlichen, wird zusätzlich auch die benötigte Zeit für die Modellbildung und die Vorhersage gemessen.

Die benutzten Parameter sind wie folgt: Für CUDTP wurde *nbar* auf 80 % der Trainingsdaten im jeweiligen Durchlauf gesetzt. Weitere Zuweisungen waren $max\_depth = 4$, $max\_subdivisions = 15$, bei ODT $kmax = 3$ und $gamma = 0{,}000\,03$; sowie $k = 1$ für KNN und bei NNet $size = 3$ und $maxit = 300$. Ansonsten wurden Vorgabewerte genutzt. Nur für den ersten Versuch (Tabelle 7.7) wurden die speziellen Werte $kmax = 2$, $max\_depth = 2$ und $max\_subdivisions = 4$ zugeteilt.

Beim ersten Vergleich (Tabelle 7.8) mit 5 Hauptkomponenten schneiden CUDTP und ODT bzgl. der ausbalancierten Genauigkeit mit jeweils 61,8 % am schlechtesten ab, da sie zwar eine hohe Sensitivität erreichen, aber nur eine geringe Spezifität. Bis auf Tree erzielen die restlichen Verfahren bei der ausbalancierten Genauigkeit Werte um 65 %, wobei NNet mit 65,4 % den höchsten Wert hat. Die höchste Spezifität erreicht hier KNN mit 51,1 % im Vergleich zum niedrigsten Wert von 36,3 % bei CUDTP und ODT. Die beste Genauigkeit erlangt SVM mit 69,1 %. Das ist nicht sehr hoch, da eine Prävalenz von 60 % vorliegt.

In der Praxis spielt auch die Zeit eine wichtige Rolle, die zur Erstellung eines Klassifikators benötigt wird, aber auch wie lange dieser für eine Vorhersage der Klasse braucht. Tabelle 7.9 zeigt dabei die durchschnittliche Zeit die das jeweilige Verfahren zur Modellbildung und Vorhersage auf den fünf Trainings- und Testmengen der Kreuzvalidierung benötigt hat.

Da wir ungefähr 32 000 Beobachtungen zur Verfügung haben, sind in der Trainingsmenge ca. 25 600 und in der Testmenge ca. 6 400 Beobachtungen enthalten. Die einfachen Verfahren Tree, Logit, LDA und QDA bilden den Klassifikator dabei in unter 1 Sekunde. KNN fällt hierbei heraus und benötigt für das Modell keine Zeit, da die k-nächsten-Nachbarn erst zur Vorhersage gebildet werden. CUDTP, ODT und NNet bleiben dabei unter 10 Sekunden. Nur RF und SVM weichen davon stärker ab mit 27 bzw. 163 Sekunden. Selbst das sind in der Praxis ertragbare Werte. Die Vorhersage für die 6 400 Testbeobachtungen reicht hingegen von einigen Millisekunden bis zu mehreren Sekunden bei SVM und KNN. Je nach Kriterium würde also eine andere Klassifikationsmethode zum Einsatz kommen.

Im nächsten Fall haben wir nur die Parameter von CUDTP und ODT verändert, um eine bessere adaptive Anpassung zu erlangen. In Tabelle 7.11 zeigen sich deshalb nur leichte Veränderungen zu Tabelle 7.8. Die Genauigkeit der Verfahren ist bei beiden von 67 % auf 67,5 % für CUDTP und auf 68,1 % für ODT gestiegen. Bei CUDTP ist die Sensitivität nahezu gleich geblieben. Jedoch hat sich die Spezifität erhöht, weshalb auch die ausbalancierte Genauigkeit leicht angestiegen ist. Bei ODT gab es eine stärkere Verschiebung von 87,4 % auf 81,6 % bei der Sensitivität und von 36,3 % auf 47,8 % bei der Spezifität. Damit hat ODT eine balancierte Genauigkeit

von 64,7 % und ist dadurch bei diesem Kriterium nur unwesentlich schlechter als SVM und NNet, die hier am besten abschneiden. Allerdings wurden diese besseren Ergebnisse mit bedeutend höherem Zeitaufwand für die Modellbildung erkauft. Für CUDTP stieg die benötigte Zeit von ca. 2 Sekunden auf ca. 1000 Sekunden. Das stellt eine Vervielfältigung der verbrauchten Zeit um den Faktor 500 dar und das bei einem Zugewinn an Genauigkeit von gerade einmal 0,5 %. Bei ODT wächst der Zeitverbrauch von ca. 7 Sekunden auf ca. 220 Sekunden. Hier liegt der Faktor also bei etwa 30 und der Genauigkeitszuwachs beträgt im Vergleich zu CUDTP sogar 1,1 %.

Da die Parameter für die anderen Verfahren gleich geblieben sind, sollten sich die Zeiten kaum unterscheiden. Für sie ergab sich eine geringfügige Verringerung der benötigten Zeit, außer bei SVM, wo eine große Schwankung von ca. 30 Sekunden bei der Modellbildung auftrat. Diese Schwankung lässt sich durch den Versuchsaufbau nicht erklären, da die gebildeten Modelle von SVM die gleichen Ergebnisse liefern, wie ein Vergleich von Tabelle 7.7 und Tabelle 7.10 zeigen. Trotz dessen lassen sich die Größenverhältnisse gut ablesen und geben einen ungefähren Eindruck über den Zeitverbrauch im Vergleich zu den anderen Verfahren.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	16851	8189	16851	8189	14646	6590	16216	6967	16258	7002	16217	7068	16570	7219	15237	6347	14992	6285	15953	6670
2	2429	4664	2429	4664	4634	6263	3064	5886	3022	5851	3063	5785	2710	5634	4043	6506	4288	6568	3327	6183

Tabelle 7.7: Aufsummierte Kontingenztafeln für einen Datensatz aus der Praxis bei 5-facher Kreuzvalidierung mit 5 Hauptkomponenten.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	0,670	0,670	0,651	0,688	0,688	0,685	0,691	0,677	0,671	0,689
Sensitivität	0,874	0,874	0,760	0,841	0,843	0,841	0,859	0,790	0,778	0,827
Spezifität	0,363	0,363	0,487	0,458	0,455	0,450	0,438	0,506	0,511	0,481
Positiver Vorhersagewert	0,673	0,673	0,690	0,699	0,699	0,696	0,697	0,706	0,705	0,705
Negativer Vorhersagewert	0,658	0,658	0,575	0,658	0,659	0,654	0,675	0,617	0,605	0,650
Prävalenz	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600
Erkennungsrate	0,524	0,524	0,456	0,505	0,506	0,505	0,516	0,474	0,467	0,496
Erkannte Prävalenz	0,779	0,779	0,661	0,721	0,724	0,725	0,740	0,672	0,662	0,704
Ausbalancierte Genauigkeit	0,618	0,618	0,623	0,650	0,649	0,646	0,649	0,648	0,644	0,654

Tabelle 7.8: Kenngrößen zu den Kontingenztafeln aus Tabelle 7.7 zur Beurteilung der Klassifikatoren.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
verbrauchte Zeit zur Modellbildung	1,954	7,166	0,217	0,180	0,080	0,075	162,774	26,957	0,000	3,562
verbrauchte Zeit zur Vorhersage	0,019	0,073	0,141	0,014	0,025	0,024	9,447	0,993	1,830	0,015

Tabelle 7.9: Durchschnittlich vergangene Zeit zur Bildung des jeweiligen Modells und die Vorhersage für neue Werte in Sekunden. Gehört zu Tabelle 7.7 und Tabelle 7.8.

Der letzte Test wurde bei Verwendung von 13 Hauptkomponenten durchgeführt und offenbart Probleme für CUDTP und ODT in der Anwendung bei höherdimensionalen Problemen. Die besten Methoden sind hier Logit, LDA, QDA, SVM, RF und NNet mit ca. 70 % ausbalancierter Genauigkeit im Gegensatz zu CUDTP mit 63,7 % und ODT mit 64,3 %. Die Verfahren mit bester Genauigkeit liefern Werte zwischen 72 % und 73 %. Hingegen erreicht CUDTP nur 68,2 % und ODT nur 68,5 %.

Logit, LDA und QDA benötigen immer noch weniger als 1 Sekunde zur Modellbildung. SVM erreicht jetzt 193 Sekunden, also etwas weniger als eine Verdopplung der Zeit, bei fast dreifacher Dimension. Zu einer Verdopplung der Zeit kam es bei RF von ca. 26 Sekunden auf ca. 57 Sekunden und bei NNet von ca. 3 Sekunden auf ca. 7 Sekunden.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	16834	7995	15734	6711	14646	6590	16216	6967	16258	7002	16217	7068	16570	7219	15237	6347	14992	6285	15953	6670
2	2446	4858	3546	6142	4634	6263	3064	5886	3022	5851	3063	5785	2710	5634	4043	6506	4288	6568	3327	6183

Tabelle 7.10: Aufsummierte Kontingenztafeln für einen Datensatz aus der Praxis bei 5-facher Kreuzvalidierung mit 5 Hauptkomponenten und anderen Parametern bei CUDTP und ODT.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	0,675	0,681	0,651	0,688	0,688	0,685	0,691	0,677	0,671	0,689
Sensitivität	0,873	0,816	0,760	0,841	0,843	0,841	0,859	0,790	0,778	0,827
Spezifität	0,378	0,478	0,487	0,458	0,455	0,450	0,438	0,506	0,511	0,481
Positiver Vorhersagewert	0,678	0,701	0,690	0,699	0,699	0,696	0,697	0,706	0,705	0,705
Negativer Vorhersagewert	0,665	0,634	0,575	0,658	0,659	0,654	0,675	0,617	0,605	0,650
Prävalenz	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600
Erkennungsrate	0,524	0,490	0,456	0,505	0,506	0,505	0,516	0,474	0,467	0,496
Erkannte Prävalenz	0,773	0,699	0,661	0,721	0,724	0,725	0,740	0,672	0,662	0,704
Ausbalancierte Genauigkeit	0,626	0,647	0,623	0,650	0,649	0,646	0,649	0,648	0,644	0,654

Tabelle 7.11: Kenngrößen zu den Kontingenztafeln aus Tabelle 7.10 zur Beurteilung der Klassifikatoren.

Die Parameter für CUDTP und ODT wurden hier so gewählt, dass die verbrauchte Zeit in der Größenordnung von SVM liegt und sind gleich denen aus Versuch 1. Diese Steuergrößen beschränken die Anzahl der möglichen Klassifikatoren sehr stark. Stellt man die Zeiten aus Tabelle 7.9 und Tabelle 7.15 gegenüber, so hat man bei der Modellbildung für CUDTP ca. 2 Sekunden im Vergleich zu ca. 210 Sekunden. Die Zeit, die benötigt wird, einen entsprechenden Klassifizierer zu finden, wächst also mit der Größe der Dimension sehr schnell. Das lässt sich auch für ODT bestätigen mit ca. 7 Sekunden gegenüber ca. 320 Sekunden für 5 Hauptkomponenten gegenüber 13.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
verbrauchte Zeit zur Modellbildung	997,303	221,305	0,216	0,176	0,079	0,075	134,092	25,986	0,000	3,462
verbrauchte Zeit zur Vorhersage	0,042	0,263	0,141	0,014	0,025	0,024	7,410	0,984	1,787	0,014

Tabelle 7.12: Durchschnittlich vergangene Zeit zur Bildung des jeweiligen Modells und die Vorhersage für neue Werte in Sekunden. Gehört zu Tabelle 7.10 und Tabelle 7.11.

Methode	CUDTP		ODT		Tree		Logit		LDA		QDA		SVM		RF		KNN		NNet	
Referenz	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
Vorhersage																				
1	16604	7550	16444	7295	14646	6590	16165	5644	16243	5779	15178	5031	16667	6069	15726	5366	15617	5691	16084	5538
2	2676	5303	2836	5558	4634	6263	3115	7209	3037	7074	4102	7822	2613	6784	3554	7487	3663	7162	3196	7315

Tabelle 7.13: Aufsummierte Kontingenztafeln für einen Datensatz aus der Praxis bei 5-facher Kreuzvalidierung mit 13 Hauptkomponenten.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
Treffergenauigkeit	0,682	0,685	0,651	0,727	0,726	0,716	0,730	0,722	0,709	0,728
Sensitivität	0,861	0,853	0,760	0,838	0,842	0,787	0,864	0,816	0,810	0,834
Spezifität	0,413	0,432	0,487	0,561	0,550	0,609	0,528	0,583	0,557	0,569
Positiver Vorhersagewert	0,687	0,693	0,690	0,741	0,738	0,751	0,733	0,746	0,733	0,744
Negativer Vorhersagewert	0,665	0,662	0,575	0,698	0,700	0,656	0,722	0,678	0,662	0,696
Prävalenz	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600	0,600
Erkennungsrate	0,517	0,512	0,456	0,503	0,505	0,472	0,519	0,489	0,486	0,501
Erkannte Prävalenz	0,752	0,739	0,661	0,679	0,685	0,629	0,708	0,656	0,663	0,673
Ausbalancierte Genauigkeit	0,637	0,643	0,623	0,700	0,696	0,698	0,696	0,699	0,684	0,702

Tabelle 7.14: Kenngrößen zu den Kontingenztafeln aus Tabelle 7.13 zur Beurteilung der Klassifikatoren.

	CUDTP	ODT	Tree	Logit	LDA	QDA	SVM	RF	KNN	NNet
verbrauchte Zeit zur Modellbildung	213,600	324,887	0,564	0,287	0,181	0,158	193,341	57,130	0,000	6,656
verbrauchte Zeit zur Vorhersage	0,042	0,139	0,136	0,017	0,028	0,031	11,417	1,139	3,794	0,020

Tabelle 7.15: Durchschnittlich vergangene Zeit zur Bildung des jeweiligen Modells und die Vorhersage für neue Werte in Sekunden. Gehört zu Tabelle 7.13 und Tabelle 7.14.

7.2.5 Zusammenfassung des Vergleichs

Die adaptiven Verfahren CUDTP und ODT sind gut für Probleme mit niedriger Dimension geeignet, da sie sich sehr flexibel an die Klassengrenzen anpassen können. In diesen Situationen ist auch der Zeitaufwand einen Klassifikator zu finden vertretbar. Jedoch wächst die Rechenintensität mit steigender Dimensionszahl sehr schnell. Deswegen sind sie in der hier implementierten Form für hochdimensionale Probleme ungeeignet.

Die beliebten Methoden SVM, RF, KNN und NNet haben auch hier wieder gezeigt, dass sie für eine Vielzahl von Problemen eingesetzt werden können, weil sie gute Ergebnisse in relativ kurzer Zeit liefern.

Auch die einfachen Strategien wie Logit, LDA und QDA sind aus der Praxis nicht wegzudenken, da sie sehr schnell anwendbar sind und in vielen Fällen eine gute Vorhersage abgeben können. Ist diese Vorhersage einmal schlecht, so kann immer noch ein anderes Verfahren ausprobiert werden, weil der Zeitaufwand gering ist.

Kurzzusammenfassung

Diese Arbeit behandelt maschinelles Lernen und führt dabei von theoretischen Überlegungen zu Implementierungen, sowie deren Vergleich mit typischen Verfahren aus der Praxis.

Nach einer kurzen Einführung in das Thema maschinelles Lernen wird der Hauptaugenmerk auf die binäre Klassifikation gelenkt. Dabei werden aufbauend auf der Wahrscheinlichkeitstheorie wichtige Begriffe wie Regressionsfunktion, Klassifikator, Bayes'scher Klassifikator, Risiko und zusätzliches Risiko eingeführt und auf deren Wechselwirkungen eingegangen. Das Ziel ist dann bei unbekannter Verteilung, anhand eines durch diese Verteilung entstanden Beobachtungsdatensatzes, einen Klassifikator zu finden, der das zusätzliche Risiko minimiert. Da die Verteilung unbekannt ist, kann man das zusätzliche Risiko nicht direkt berechnen und versucht es durch Aufspaltung in Schätz- und Näherungsfehler nach oben abzuschätzen. Das führt zur VC Dimension und einem Objekt, welches als Modulus bezeichnet wird. Unter gewissen Zusatzannahmen an die Verteilung, wie Randbedingungen und Zugehörigkeit zu einer Approximationsklasse, lässt sich dann die Abschätzung der Fehler bewerkstelligen. Jedoch sind die Parameter in diesen Bedingungen nicht bekannt und es stellt sich die Frage, wie man trotzdem eine möglichst günstige Abschätzung erhält. Das führt zu einer speziellen Modellwahl, die für den ausgewählten Klassifikator eine ebenso gute Schranke liefert, wie wenn man die Wahl unter Kenntnis der unbekannten Parameter treffen würde. Dieses Wissen wird dann auf dyadische Bäume und deren Partitionierungen angewendet. Darauf aufbauend wird ein Baumalgorithmus implementiert, der diese Modellauswahl benutzt und zusätzlich ein Vergleichsalgorithmus der ebenfalls dyadische Bäume gebraucht. Anschließend folgt eine Einführung in typische praxisrelevante Methoden zur Klassifizierung und der Vergleich mit den implementierten Verfahren mittels der Programmiersprache und Softwareumgebung für statistische Berechnungen R. Dabei liefern meist mehrere der gewöhnlicherweise verwendeten Verfahren sehr gute Ergebnisse. Außerdem zeigt sich, dass die dyadischen Bäume für niedrigdimensionale Probleme gute Ergebnisse erzielen und für hochdimensionale Problemstellungen sehr rechenintensiv und damit zeitintensiv werden.

Insgesamt liefert die Diplomarbeit damit einen praxisnahen und theoretisch fundierten Einstieg in das Thema des maschinellen Lernens mit anwendungsorientierten Beispielen in der Programmiersprache R.

Literatur

[Beh13] E. Behrends. *Elementare Stochastik. Ein Lernbuch - von Studierenden mitentwickelt*. 1. Aufl. Vieweg+Teubner Verlag, 2013. DOI: 10.1007/978-3-8348-2331-1. URL: http://www.springer.com/de/book/9783834819390.

[Bin+14a] P. Binev, A. Cohen, W. Dahmen und R. DeVore. „Classification algorithms using adaptive partitioning“. In: *The Annals of Statistics* 42.6 (2014), S. 2141–2163. DOI: 10.1214/14-AOS1234.

[Bin+14b] P. Binev, A. Cohen, W. Dahmen und R. DeVore. „Supplement to »Classification algorithms using adaptive partitioning«“. In: *The Annals of Statistics* 42.6 (2014), S. 2141–2163. DOI: 10.1214/14-AOS1234SUPP.

[Bla+07] G. Blanchard, C. Schäfer, Y. Rozenholc und K.-R. Müller. „Optimal dyadic decision trees“. In: *Machine Learning* 66 (2007), S. 209–241. DOI: 10.1007/s10994-007-0717-6.

[BM06] P. L. Bartlett und S. Mendelson. „Empirical minimization“. In: *Probability Theory and Related Fields* 135 (2006), S. 311–334. DOI: 10.1007/s00440-005-0462-3.

[Bre+84] L. Breiman, J. Friedman, C. Stone und R. Olshen. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984. URL: http://books.google.de/books?id=JwQx-WOmSyQC.

[Dah16] D. B. Dahl. *xtable: Export Tables to LaTeX or HTML*. R package version 1.8-2. 2016. URL: https://CRAN.R-project.org/package=xtable.

[Dei14] A. Deitmar. *Analysis*. 1. Aufl. Springer Spektrum, 2014. DOI: 10.1007/978-3-642-54810-9. URL: http://www.springer.com/de/book/9783642548093.

[DGL96] L. Devroye, L. Györfi und G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Bd. 31. Springer-Verlag New York, 1996. DOI: 10.1007/978-1-4612-0711-5.

[Edd13] D. Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. 1. Aufl. Springer-Verlag New York, 2013. DOI: 10.1007/978-1-4614-6868-4. URL: http://www.springer.com/de/book/9781461468677.

[EF11] D. Eddelbuettel und R. François. „Rcpp: Seamless R and C++ Integration“. In: *Journal of Statistical Software* 40 (2011), S. 1–18. DOI: 10.1007/978-1-4614-6868-4. URL: http://www.jstatsoft.org/v40/i08/.

[Ern00] H. Ernst. *Grundlagen und Konzepte der Informatik*. Vieweg + Teubner Verlag, 2000. DOI: 10.1007/978-3-663-10229-8.

[Gyö+02] L. Györfi, M. Kohler, A. Krzyżak und H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. 1. Aufl. Springer-Verlag New York, 2002. DOI: 10.1007/b97848. URL: http://www.springer.com/de/book/9781441929983.

[Hil06] S. Hildebrandt. *Analysis 1*. 2. Aufl. Springer-Verlag Berlin Heidelberg, 2006. DOI: 10.1007/3-540-29285-3. URL: http://www.springer.com/de/book/9783540253686.

[HTF09] T. Hastie, R. Tibshirani und J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction, Second Edition*. 2. Aufl. Springer-Verlag New York, 2009. DOI: 10.1007/978-0-387-84858-7. URL: http://statweb.stanford.edu/~tibs/ElemStatLearn/.

[Jam+16] G. James, D. Witten, T. Hastie und R. Tibshirani. *An Introduction to Statistical Learning. With Applications in R*. 6. Aufl. Springer-Verlag New York, Jan. 2016. DOI: 10.1007/978-1-4614-7138-7. URL: http://www-bcf.usc.edu/~gareth/ISL/.

[Kle13] A. Klenke. *Wahrscheinlichkeitstheorie*. 3. Aufl. Springer Spektrum, 2013. DOI: 10.1007/978-3-642-36018-3. URL: http://www.springer.com/de/book/9783642360176.

[Knu05] D. E. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 3: Generating All Combinations and Partitions*. Addison-Wesley Professional, 2005. URL: http://www-cs-faculty.stanford.edu/~uno/taocp.html.

[Kol11] V. Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems. École d'Été de Probabilités de Saint-Flour XXXVIII-2008*. 1. Aufl. Springer-Verlag Berlin Heidelberg, 2011. DOI: 10.1007/978-3-642-22147-7. URL: http://www.springer.com/de/book/9783642221460.

[Kuh+16] M. Kuhn u. a. *caret: Classification and Regression Training*. R package version 6.0-64. 2016. URL: https://CRAN.R-project.org/package=caret.

[Lan15] B. Lantz. *Machine Learning with R - Second Edition*. 2. Aufl. Packt Publishing, Juli 2015. URL: http://amazon.de/o/ASIN/1784393908/.

[LW02] A. Liaw und M. Wiener. „Classification and Regression by randomForest“. In: *R News* 2.3 (2002), S. 18–22. URL: http://CRAN.R-project.org/doc/Rnews/.

[Mey+15] D. Meyer u. a. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.6-7. 2015. URL: https://CRAN.R-project.org/package=e1071.

[Rip16] B. Ripley. *tree: Classification and Regression Trees*. R package version 1.0-37. 2016. URL: https://CRAN.R-project.org/package=tree.

[RUL14] A. Rajaraman, J. D. Ullman und J. Leskovec. *Mining of Massive Datasets*. 2. Aufl. Cambridge University Press, 2014. URL: http://www.mmds.org/.

[SB16] C. Sharpsteen und C. Bracken. *tikzDevice: R Graphics Output in LaTeX Format*. R package version 0.10. 2016. URL: https://CRAN.R-project.org/package=tikzDevice.

[VR02] W. Venables und B. Ripley. *Modern Applied Statistics with S*. 4. Aufl. Springer-Verlag New York, 2002. DOI: 10.1007/978-0-387-21706-2. URL: http://www.stats.ox.ac.uk/pub/MASS4/.

[vW96] A. van der vaart und J. Wellner. *Weak Convergence and Empirical Processes. With Applications to Statistics*. 1. Aufl. Springer-Verlag New York, 1996. DOI: 10.1007/978-1-4757-2545-2. URL: http://www.springer.com/de/book/9781475725476.

[WDE15] H. Wickham, P. Danenberg und M. Eugster. *roxygen2: In-Source Documentation for R*. R package version 5.0.1. 2015. URL: https://CRAN.R-project.org/package=roxygen2.

[Wic09] H. Wickham. *ggplot2. Elegant Graphics for Data Analysis*. 1. Aufl. Springer-Verlag New York, 2009. DOI: 10.1007/978-0-387-98141-3. URL: http://www.springer.com/de/book/9780387981413.

[Wic11] H. Wickham. „testthat: Get Started with Testing“. In: *The R Journal* 3 (2011), S. 5–10. URL: http://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf.

[Xie14] Y. Xie. „knitr: A Comprehensive Tool for Reproducible Research in R“. In: *Implementing Reproducible Computational Research*. Hrsg. von V. Stodden, F. Leisch und R. D. Peng. Chapman und Hall/CRC, 2014. URL: http://www.crcpress.com/product/isbn/9781466561595.

[Xie15] Y. Xie. *Dynamic Documents with R and knitr*. 2nd. Boca Raton, Florida: Chapman und Hall/CRC, 2015. URL: http://yihui.name/knitr/.

[Xie16] Y. Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.12.3. 2016. URL: https://cran.r-project.org/web/packages/knitr/index.html.

[R C15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2015. URL: https://www.R-project.org/.

Abbildungsverzeichnis

1 Beispiel zu Überanpassung 4

2 Baum, der bei $d = 2$ und $m = 4$ zur Bitfolge 1010 0000 0100 0000 führt. 42

3 Würfel und deren Geschwister 45
4 Belegungsbaum zu Abbildung 3 mit Höhe 4. 46
5 Vollständiger Belegungsbaum zu Abbildung 3 mit Höhe 3. 46

6 Lineare Regression Airpassengerdatensatz in R. 59
7 LDA angewendet auf den Iris Datensatz 63
8 Beispiel für Maximal Margin Klassifizierer 66
9 Support Vector Klassifizierer. 67
10 Voronoi-Diagramm. 71
11 Aufbau neuronaler Netze. 72
12 Sigmoidfunktionen. 73
13 5-fache Kreuzvalidierung 74
14 Kreis CUDTP 77
15 Kreis ODT 78
16 Schachbrettmuster CUDTP 78
17 Schachbrettmuster ODT 79
18 diagonale Fläche CUDTP 80
19 diagonale Fläche ODT 80

Tabellenverzeichnis

7.1 Kontingenztafeln Kreis 76
7.2 Kenngrößen Kreis 76
7.3 Kontingenztafeln Schachbrettmuster 81
7.4 Kenngrößen Schachbrettmuster 81
7.5 Kontingenztafeln diagonale Fläche 81
7.6 Kenngrößen diagonale Fläche 81
7.7 Aufsummierte Kontingenztafeln bei 5 Hauptkomponenten 84
7.8 Kenngrößen bei 5 Hauptkomponenten 84
7.9 Vergange Zeit bei 5 Hauptkomponenten 84
7.10 Kontingenztafeln bei 5 Hauptkomponenten andere Parameter 85
7.11 Kenngrößen bei 5 Hauptkomponenten andere Parameter 85
7.12 Vergange Zeit bei 5 Hauptkomponenten andere Parameter 86
7.13 Aufsummierte Kontingenztafeln bei 13 Hauptkomponenten 86
7.14 Kenngrößen bei 13 Hauptkomponenten 86
7.15 Vergange Zeit bei 13 Hauptkomponenten 86

B.1 Kontingenztafel zur Diagnoseprüfung 113
B.2 Kenngrößentabelle mit Formeln 113

Programmauszüge

6.1 Parameter der Hauptfunktion. CUDTP.cpp 48
6.2 Berechne den Vektor l_j^*. CUDTP.cpp 49
6.3 Durchlaufe alle möglichen Kombinationen für l_j^*. CUDTP.cpp 51
6.4 Durchlauf aller Kombinationen für l_j^* bei variabler Maximalanzahl . . 51
6.5 Generierung und Speicherung der Zellen. ODT.cpp 54
6.6 Speichere die trivialen Partitionen ab. ODT.cpp 55
6.7 Finde Geschwister- und Vorgängerzellen. ODT.cpp 56
6.8 Bestimme optimale Teilpartitionen. ODT.cpp 57

Liste der Algorithmen

1 Lexikographische Kombinationen 50
2 Lexikon-basierender ODT Algorithmus 53

A | Abschätzungen und Näherungen

Die hier vorgenommenen Definitionen in A.1 stammen aus [Kle13, S. 93] und [Kle13, S. 145 f.].

Definition A.1. (L^p-Räume und L^p-Normen) Sei $(\Omega, \mathfrak{A}, \mu)$ ein Maßraum. Für messbares $f\colon \Omega \to \overline{\mathbb{R}}$ mit $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ definieren wir

$$\|f\|_p := \|f\|_{\mu,p} := \left(\int |f|^p \, d\mu \right)^{1/p}, \qquad \text{falls } p \in [1, \infty)$$

und

$$\|f\|_\infty := \|f\|_{\mu,\infty} := \inf\{K \geq 0 \colon \mu(\{|f| > K\}) = 0\}.$$

Ferner definieren wir für jedes $p \in [1, \infty]$ den Vektorraum

$$\mathcal{L}^p(\mu) := \mathcal{L}^p(\Omega, \mathfrak{A}, \mu) := \{f\colon \Omega \to \overline{\mathbb{R}} \mid f \text{ ist messbar und } \|f\|_p < \infty\}.$$

Für jedes $p \in [1, \infty]$ definieren wir mit

$$\mathcal{N} := \{f\colon \Omega \to \overline{\mathbb{R}} \mid f \text{ ist messbar und } \mu\text{-fast überall gilt } f = 0\}$$

den Quotientenraum

$$L^p(\mu) := L^p(\Omega, \mathfrak{A}, \mu) := \mathcal{L}^p(\Omega, \mathfrak{A}, \mu)/\mathcal{N} = \{[f] := f + \mathcal{N} := f \in \mathcal{L}^p(\mu)\}$$

und setzen

$$\|[f]\|_p := \|[f]\|_{\mu,p} := \|f\|_{\mu,p}$$

für $[f] \in L^p(\mu)$ und ein $\tilde{f} \in [f]$, sowie

$$\int [f] \, d\mu := \int \tilde{f} \, d\mu,$$

falls dieser Ausdruck für $\tilde{f}$ definiert ist. Andere Schreibweisen für $\|[f]\|_{\mu,p}$ sind auch $\|[f]\|_{L_p(\mu)}$ oder $\|[f]\|_{L^p(\mu)}$.

Das nachfolgende Lemma stammt aus [BM06, S. 6].

Lemma A.2. ***(Talagrand's Konzentrationsungleichung)*** *Sei $\mathcal{F}$ eine Klasse von Funktionen definiert auf $\mathbf{X}$ und $\mathbf{P}$ ein Wahrscheinlichkeitsmaß, so dass für jedes $f \in \mathcal{F}$ die Bedingungen $\|f\|_\infty \leq b$ und $\mathbf{E}[f] = 0$ gelten. Seien $X_1, \ldots, X_n$*

unabhängige, $\mathbf{P}$*-verteilte Zufallsvariablen und setze* $\sigma^2 = n \sup_{f \in \mathcal{F}} \mathbf{Var}[f]$*. Definiert man nun*

$$Z = \sup_{f \in \mathcal{F}} \sum_{i=1}^{n} f(X_i)$$

und

$$\bar{Z} = \sup_{f \in \mathcal{F}} \left| \sum_{i=1}^{n} f(X_i) \right|,$$

so ergibt sich für jedes $x > 0$*, dann*

$$\mathbf{P}\left[|Z - \mathbf{E}[Z]| \geq x\right] \leq C \exp\left(-\frac{x}{Kb} \ln\left(1 + \frac{bx}{\sigma^2 + b\mathbf{E}[\bar{Z}]}\right)\right). \tag{A.1}$$

Dabei sind C *und* K *absolute Konstanten. Die gleiche Ungleichung gilt ebenfalls, wenn* Z *durch* $\bar{Z}$ *in* (A.1) *ersetzt wird.*

Die im Anschluss festgelegten Bezeichnungen aus [Kol11, S. 1 ff.] spielen für den Beweis von Satz 3.1 mittels der nachstehenden Lemmata eine wichtige Rolle.

Definition A.3. Seien $X, X_1, \ldots, X_n$ auf dem Wahrscheinlichkeitsraum $(\Omega, \Sigma, \mathbf{P})$ unabhängig und identisch verteilte Zufallsvariablen mit Werten in einem messbaren Raum $(S, \mathfrak{A})$ mit gemeinsamer Verteilung P. Ferner bezeichne P_n das empirische Maß, welches auf der Stichprobe $(X_1, \ldots, X_n)$ der ersten n Beobachtungen basiert:

$$P_n := \frac{1}{n} \sum_{k=1}^{n} \delta_{X_k}.$$

Dabei ist $\delta_x, x \in S$ das Diracmaß. Außerdem sei $\mathcal{F}$ eine Menge von messbaren Funktionen $f \colon S \to \mathbb{R}$. Dann sei

$$\mathbf{E}[f(X)] = \int_S f \, \mathrm{d}P = Pf,$$

$$\frac{1}{n} \sum_{k=1}^{n} f(X_k) = \int_S f \, \mathrm{d}P_n = P_n f,$$

$$\|Y\|_{\mathcal{F}} := \sup_{f \in \mathcal{F}} |Y(f)|, \qquad Y \colon \mathcal{F} \to \mathbb{R},$$

$$R_n(f) := \frac{1}{n} \sum_{k=1}^{n} \epsilon_k f(X_k), \qquad f \in \mathcal{F},$$

mit $\epsilon_1, \ldots, \epsilon_n$ sind unabhängig und identisch verteilte Rademacher Zufallsvariablen, d.h. sie sind symmetrische Bernoulli Zufallsvariablen mit Werten $+1$ und -1 jeweils mit Wahrscheinlichkeit 0,5 und sind auch unabhängig von den Daten $(X_1, \ldots, X_n)$.

Die Definition von $N(T, d, \varepsilon)$ wurde in [Kol11, S. 32] vorgenommen:

Definition A.4. Für einen pseudo-metrischen Raum (T, d) und eine Konstante $\varepsilon > 0$ wird mit $N(T, d, \varepsilon)$ die ε-Überdeckungszahl von (T, d) bezeichnet. Das ist die kleinste Zahl von offenen Bällen mit Radius ε, die benötigt wird, um T zu überdecken.

Das nächste Lemma ist eine Folgerung von Theorem 3.12 aus [Kol11, S. 41], wobei bei Betrachtung des Beweises in (3.11) wegen den Integraleigenschaften eine schwächere Forderung an ε gestellt werden kann. Diese Folgerung besteht dabei aus den Ungleichungen (3.16) und (3.17) aus [Kol11, S. 44]. Die Definitionen von U und F stammen aus [Kol11, S. 40].

Definition A.5. Sei X eine Menge. Eine Funktion $F\colon X \to \mathbb{R}$ heißt Einhüllende für eine Menge von Funktionen $\mathcal{F}$, falls für alle $f \in \mathcal{F}$ mit $f\colon X \to \mathbb{R}$ das Folgende gilt:

$$\forall x \in X\colon |f(x)| \leq F(x).$$

Lemma A.6. *Sei* $\sigma^2 := \sup_{f\in\mathcal{F}} Pf^2$, $\sigma_n^2 := \sup_{f\in\mathcal{F}} P_n f^2$, *$F$ eine messbare Einhüllende von $\mathcal{F}$ und $U > 0$ ein Konstante mit $F(x) \leq U$ für alle $x \in X$. Wenn ein $A > 0$ und ein $V > 0$ existiert, sodass für alle bis auf endlich viele ε mit $0 < \varepsilon < 2\sigma_n$*

$$N(\mathcal{F}, L_2(P_n), \varepsilon) \leq \left(\frac{A \|F\|_{L_2(P_n)}}{\varepsilon} \right)^V$$

gilt, so existiert eine universelle Konstante $C > 0$ für $\sigma^2 \geq \frac{c}{n}$ mit einer Konstanten c und es ist

$$\mathbf{E} \|R_n\|_{\mathcal{F}} \leq C \max \left\{ \sigma \sqrt{\frac{V}{n} \cdot \ln \left(\frac{A \|F\|_{L_2(P)}}{\sigma} \right)}; \frac{VU}{n} \cdot \ln \left(\frac{A \|F\|_{L_2(P)}}{\sigma} \right) \right\}.$$

Die Definition des Hypographen und des zugehörigen VC-Index bzw. der zugehörigen VC Dimension stammt aus [vW96, S. 141].

Definition A.7. Der Hypograph einer Funktion $f\colon X \to \mathbb{R}$ ist die folgende Teilmenge von $X \times \mathbb{R}$:

$$\{(x, t)\colon t < f(x)\}.$$

Eine Menge von Funktionen $\mathcal{F}$ wird eine VC-Hypographen Klasse genannt, falls das Mengensystem

$$\mathcal{F}^+ := \left\{ \{(x, t)\colon t < f(x)\} : f \in \mathcal{F} \right\} \tag{A.2}$$

aller Hypographen von $\mathcal{F}$ endliche VC Dimension hat. Man setzt dann

$$V_{\mathcal{F}} := V_{\mathcal{F}^+}.$$

Folgendes Lemma zeigt einfache Ergebnisse für Splitterkoeffizienten von Mengensystemen, die durch Kombinationen von Mengen aus mehreren Mengensystemen entstehen. Es ist mit Beweis in [DGL96, S. 219] zu finden und wird später benutzt, um einen Zusammenhang zwischen der VC Dimensionen $\mathcal{S}_k$ und der VC Dimension der Klasse der Hypographen (A.2) herzustellen. Die Hypographen werden durch die Menge der Funktionen $\mathcal{F}_k$ gebildet, wobei die Menge $\mathcal{F}_k$ durch (3.10) von $\mathcal{S}_k$ abhängt.

Lemma A.8. *Seien $\mathcal{A}, \mathcal{A}_1, \mathcal{A}_2, \widehat{\mathcal{A}}, \widetilde{\mathcal{A}}$ Mengensysteme und $l \in \mathbb{N}$. So gilt:*

1. *Für $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ gilt $s(\mathcal{A}, l) \leq s(\mathcal{A}_1, l) + s(\mathcal{A}_2, l)$.*
2. *Sei $\mathcal{A}_C := \left\{A^C \colon A \in \mathcal{A}\right\}$. Dann ist $s(\mathcal{A}_C, l) = s(\mathcal{A}, l)$.*
3. *Für $\mathcal{A} = \left\{\widehat{A} \cap \widetilde{A} \colon \widehat{A} \in \widehat{\mathcal{A}}, \widetilde{A} \in \widetilde{\mathcal{A}}\right\}$ ist $s(\mathcal{A}, l) \leq s(\widehat{\mathcal{A}}, l) s(\widetilde{\mathcal{A}}, l)$.*
4. *Für $\mathcal{A} = \left\{\widehat{A} \cup \widetilde{A} \colon \widehat{A} \in \widehat{\mathcal{A}}, \widetilde{A} \in \widetilde{\mathcal{A}}\right\}$ ist $s(\mathcal{A}, l) \leq s(\widehat{\mathcal{A}}, l) s(\widetilde{\mathcal{A}}, l)$.*
5. *Für $\mathcal{A} = \left\{\widehat{A} \times \widetilde{A} \colon \widehat{A} \in \widehat{\mathcal{A}}, \widetilde{A} \in \widetilde{\mathcal{A}}\right\}$ ist $s(\mathcal{A}, l) \leq s(\widehat{\mathcal{A}}, l) s(\widetilde{\mathcal{A}}, l)$.*

Die nächsten beiden Lemmata sind hilfreich, um die Bedingung an die Überdeckungszahl aus Lemma A.6 zu erfüllen. Das zweite Lemma wird in [Kol11, S. 44] als Ungleichung (3.20) angegeben und das erste Lemma ist Theorem 2.6.7 aus [vW96, S. 141]. Dort findet man auch den Beweis. Es ist zu beachten, dass die Definition des VC Index in [vW96, S. 141] sich von der Definition der VC Dimension (2.7) unterscheidet, aber die Gleiche Notation besitzt. Da

$$V_{\mathcal{A}} = \sup\left\{l \in \mathbb{N} \colon s(\mathcal{A}, l) = 2^l\right\} = \inf\left\{l \in \mathbb{N} \colon s(\mathcal{A}, l) < 2^l\right\} - 1$$

gilt, lässt sich Theorem 2.6.7 leicht in die folgende Version unter Nutzung der VC Dimension umwandeln.

Lemma A.9. *Sei $\mathcal{F}$ eine VC-Hypographen Klasse und die $f \in \mathcal{F}$ seien messbar. Ferner sei F eine messbare Einhüllende von $\mathcal{F}$. Dann gibt es eine allgemeingültige Konstante $K > 0$, so dass für jedes Wahrscheinlichkeitsmaß Q, jedes $r \geq 1$ und $\varepsilon \in (0;1)$ mit $\|F\|_{Q,r} > 0$ die folgende Abschätzung für die ε-Überdeckungszahl gilt:*

$$N(\mathcal{F}, L_r(Q), \varepsilon \|F\|_{Q,r}) \leq K(V_{\mathcal{F}} + 1)(16e)^{V_{\mathcal{F}}+1} \left(\frac{1}{\varepsilon}\right)^{rV_{\mathcal{F}}}. \tag{A.3}$$

Lemma A.10. *Sei $\mathcal{S}$ ein Mengensystem mit $\mathcal{S} \subseteq \mathfrak{P}(\mathbf{X})$ und*

$$\mathcal{F} := \{f_S \colon \mathbf{X} \times \mathbf{Y} \to \mathbb{R} \colon S \in \mathcal{S}, f_S(x,y) = y\left(\mathbb{1}_{\Omega_S} - 1_S\right)(x)\}$$

für Mengen $\mathbf{X}, \mathbf{Y}$. Dann ist

$$V_{\mathcal{F}} \leq 2V_{\mathcal{S}}.$$

Ist F eine messbare Einhüllende von $\mathcal{F}$ und $r \geq 1$, so gibt es eine Konstante $A > 0$ abhängig von r, sodass für jedes Wahrscheinlichkeitsmaß Q, sowie $0 < \varepsilon < \|F\|_{Q,r}$ mit $\|F\|_{Q,r} > 0$ die folgende Abschätzung gilt:

$$N(\mathcal{F}, L_r(Q), \varepsilon) \leq \left(\frac{A\,\|F\|_{Q,r}}{\varepsilon}\right)^{rV_{\mathcal{F}}}.$$

Speziell erhält man mit $r = 2$ und Nutzung des empirischen Maßes P_n:

$$N(\mathcal{F}, L_2(P_n), \varepsilon) \leq \left(\frac{A\,\|F\|_{L_2(P_n)}}{\varepsilon}\right)^{2V_{\mathcal{F}}} \leq \left(\frac{A\,\|F\|_{L_2(P_n)}}{\varepsilon}\right)^{4V_{\mathcal{S}}}. \tag{A.4}$$

Beweis. Sei $\mathcal{Y} := \{\{-1\}; \{1\}\}$ und $\mathcal{T} := \{(-\infty, -1); (-\infty, 0); (-\infty, 1)\}$. Damit ist $|\mathcal{Y}| = 2$ und $|\mathcal{T}| = 3 < 4$, sowie mit Lemma 2.10 dann

$$V_{\mathcal{Y}} \leq \log_2 |\mathcal{Y}| = 1, \qquad V_{\mathcal{T}} \leq \log_2 |\mathcal{T}| < 2.$$

Also erhält man $V_{\mathcal{Y}} = V_{\mathcal{T}} = 1$. Außerdem ist

$$\mathcal{F}^+ \subseteq \mathcal{G} := \{S \times Y \times T \colon S \in \mathcal{S}, Y \in \mathcal{Y}, T \in \mathcal{T}\} \cup \{S^C \times Y \times T \colon S \in \mathcal{S}, Y \in \mathcal{Y}, T \in \mathcal{T}\}$$

und somit

$$V_{\mathcal{F}^+} \leq V_{\mathcal{G}}.$$

Außerdem folgt für $\mathcal{G}$ mit Verwendung von Lemma A.8:

$$s(\mathcal{G}, l) \leq s(\mathcal{S}, l)s(\mathcal{Y}, l)s(\mathcal{T}, l) + s(\mathcal{S}_C, l)s(\mathcal{Y}, l)s(\mathcal{T}, l) = 2s(\mathcal{S}, l)s(\mathcal{Y}, l)s(\mathcal{T}, l).$$

Daraus ergibt sich

$$V_{\mathcal{F}} = V_{\mathcal{F}^+} \leq V_{\mathcal{G}} \leq 2V_{\mathcal{S}}V_{\mathcal{Y}}V_{\mathcal{T}} = 2V_{\mathcal{S}} \cdot 1 \cdot 1 = 2V_{\mathcal{S}}.$$

Setzt man $\tilde{\varepsilon} := \varepsilon \|F\|_{Q,r}$, so folgt mit Lemma A.9 bei einsetzen von $\tilde{\varepsilon}$ in (A.3):

$$\begin{aligned} N(\mathcal{F}, L_r(Q), \tilde{\varepsilon}) &\leq K(V_\mathcal{F}+1)(16e)^{V_\mathcal{F}+1}\left(\frac{\|F\|_{Q,r}}{\tilde{\varepsilon}}\right)^{rV_\mathcal{F}} \\ &\leq K \cdot 2^{V_\mathcal{F}+1}(16e)^{V_\mathcal{F}+1}\left(\frac{\|F\|_{Q,r}}{\tilde{\varepsilon}}\right)^{rV_\mathcal{F}} \\ &= 2 \cdot 16e \cdot K \cdot \left(2^{1/r}\right)^{rV_\mathcal{F}}\left((16e)^{1/r}\right)^{rV_\mathcal{F}}\left(\frac{\|F\|_{Q,r}}{\tilde{\varepsilon}}\right)^{rV_\mathcal{F}} \\ &\leq \left((2 \cdot 16e \cdot K)^{1/r}\right)^{rV_\mathcal{F}} \cdot \left(2^{1/r}\right)^{rV_\mathcal{F}}\left((16e)^{1/r}\right)^{rV_\mathcal{F}}\left(\frac{\|F\|_{Q,r}}{\tilde{\varepsilon}}\right)^{rV_\mathcal{F}} \\ &= \left(\frac{A\|F\|_{Q,r}}{\tilde{\varepsilon}}\right)^{rV_\mathcal{F}} \end{aligned}$$

mit

$$A := (2 \cdot 16e \cdot K \cdot 2 \cdot 16e)^{1/r} = (1024eK)^{1/r}.$$

Die letzte Behauptung (A.4) erhält man dann durch Einsetzen der gerade gewonnenen Ergebnisse. □

Lemma A.11 stellt den Spezialfall von Theorem 2.1 aus [Kol11, S. 18] dar.

Lemma A.11. *Für jede Menge $\mathcal{F}$ von P-integrierbaren Funktionen gilt*

$$\mathbf{E}\|P_n - P\|_\mathcal{F} \leq 2\mathbf{E}\|R_n\|_\mathcal{F}.$$

Lemma A.12. *Seien die Bezeichnungen wie in Lemma A.6 und Lemma A.10 und*

$$Z := \sup_{f\in\mathcal{F}}\left|\sum_{k=1}^{n}(f(X_k) - \mathbf{E}[f(X)])\right|.$$

Dann ist

$$\mathbf{E}[Z] \leq 8n \cdot C \max\left\{\sigma; \sqrt{\frac{V_\mathcal{S}\ln(A\sigma^{-1})}{n}}\right\} \cdot \sqrt{\frac{V_\mathcal{S}\ln(A\sigma^{-1})}{n}}.$$

Beweis. Zuerst nutzen wir Lemma A.6 mit $F = 1_{\mathbf{X}\times\mathbf{Y}}$, $U = 1$ und $V = 4V_\mathcal{S}$, um den folgenden Erwartungswert wie folgt abzuschätzen:

$$\begin{aligned} \mathbf{E}\|R_n\|_\mathcal{F} &\leq C \max\left\{\sigma\sqrt{\frac{4V_\mathcal{S}\ln(A\sigma^{-1})}{n}}; \frac{4V_\mathcal{S}\ln(A\sigma^{-1})}{n}\right\} \\ &\leq 4C \max\left\{\sigma\sqrt{\frac{V_\mathcal{S}\ln(A\sigma^{-1})}{n}}; \frac{V_\mathcal{S}\ln(A\sigma^{-1})}{n}\right\}, \end{aligned} \tag{A.5}$$

falls für alle $0 < \varepsilon < 2\sigma_n$

$$N(\mathcal{F}, L_2(P_n), \varepsilon) \leq \left(\frac{A}{\varepsilon}\right)^{4V_{\mathcal{S}}}$$

gilt. Dabei ist

$$\sigma_n = \sqrt{\sup_{f\in\mathcal{F}} P_n f^2} = \sqrt{\sup_{S\in S} P_n f_S^2} = \sqrt{\frac{1}{n}\sum_{j=1}^{n} f_S^2(x_j, y_j)} \leq \sqrt{\frac{1}{n}\sum_{j=1}^{n} 1} \leq 1.$$

Für $\varepsilon \in (0;1)$ lässt sich die Bedingung an $N(\mathcal{F}, L_2(P_n), \varepsilon)$ durch Lemma A.10 erfüllen. Für $\varepsilon \in (1;2)$ hat man für $S \in \mathcal{S}$

$$\|f_S - 0\|^2_{L_2(P_n)} = \|f\|^2_{L_2(P_n)} = \frac{1}{n}\sum_{j=1}^{n} f^2(x_i, y_i) \leq \frac{1}{n}\sum_{j=1}^{n} 1 = 1 < \varepsilon$$

und somit $N(\mathcal{F}, L_2(P_n), \varepsilon) = 1 \leq \left(\frac{A}{\varepsilon}\right)^{4V_{\mathcal{S}}}$ für $A \geq 2$ und $\varepsilon \in (1;2)$.

Wir bringen die Norm in Definition A.3 in eine für uns brauchbare Form:

$$\begin{aligned}
\|P_n - P\|_{\mathcal{F}} &= \sup_{f\in\mathcal{F}} |(P_n - P)(f)| \\
&= \sup_{f\in\mathcal{F}} |P_n(f) - P(f)| \\
&= \sup_{f\in\mathcal{F}} \left|\frac{1}{n}\sum_{k=1}^{n} f(X_k) - \mathbf{E}[f(X)]\right| \\
&= \frac{1}{n}\sup_{f\in\mathcal{F}} \left|\sum_{k=1}^{n} (f(X_k) - \mathbf{E}[f(X)])\right| \\
&= \frac{Z}{n}.
\end{aligned} \tag{A.6}$$

Wendet man nun unsere Umformung (A.6), sowie Lemma A.11 und die Abschätzung (A.5) für den Erwartungswert von Z an, so ergibt sich

$$\begin{aligned}
\mathbf{E}[Z] &\overset{(A.6)}{=} n\mathbf{E}\,\|P_n - P\|_{\mathcal{F}} \\
&\leq 2n\mathbf{E}\,\|R_n\|_{\mathcal{F}} \\
&\overset{(A.5)}{\leq} 2n \cdot 4C \max\left\{\sigma\sqrt{\frac{V_{\mathcal{S}}\ln(A\sigma^{-1})}{n}}; \frac{V_{\mathcal{S}}\ln(A\sigma^{-1})}{n}\right\} \\
&= 8n \cdot C \max\left\{\sigma; \sqrt{\frac{V_{\mathcal{S}}\ln(A\sigma^{-1})}{n}}\right\} \cdot \sqrt{\frac{V_{\mathcal{S}}\ln(A\sigma^{-1})}{n}}.
\end{aligned}$$

□

Hier eine leicht abgewandelte Version der Markov'schen Ungleichung aus [Kle13, S. 110] in der Form, wie sie für den Beweis der Ungleichung von Bennett benötigt wird.

Lemma A.13. (Markov'sche Ungleichung) *Sei X eine reelle Zufallsvariable und $f\colon \mathbb{R} \to [0,\infty)$ monoton wachsend. Dann gilt für jedes $\varepsilon > 0$ mit $f(\varepsilon) > 0$*

$$\mathbf{P}\left[X \geq \varepsilon\right] \leq \frac{\mathbf{E}\left[f(X)\right]}{f(\varepsilon)}.$$

Beweis. Für f monoton wachsend, gilt $\{f(X) \geq f(\varepsilon)\} = \{X \geq \varepsilon\}$ und unter Verwendung der Eigenschaften des Erwartungswertes erhält man

$$\begin{aligned} E[f(X)] &\geq E\left[f(X)\mathbb{1}_{\{f(X)\geq f(\varepsilon)\}}\right] \\ &\geq E\left[f(\varepsilon)\mathbb{1}_{\{f(X)\geq f(\varepsilon)\}}\right] \\ &= f(\varepsilon)E\left[\mathbb{1}_{\{X\geq\varepsilon\}}\right] \\ &= f(\varepsilon)P[X \geq \varepsilon]. \end{aligned}$$

□

Für die nachfolgenden Lemmata und Beweise vergleiche man mit [DGL96, S. 124] und [DGL96, S. 130].

Lemma A.14. (Ungleichung von Bennett) *Seien $X_1, \ldots, X_n$ unabhängige reelle Zufallsvariablen mit $\mathbf{E}[X_i] = 0$ und es gelte $|X_i| \leq c$ fast sicher. Sei*

$$\sigma^2 := \frac{1}{n}\sum_{k=1}^{n} \mathbf{Var}[X_k].$$

Dann gilt für jedes $\varepsilon > 0$

$$\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] \leq \exp\left(-\frac{n\sigma^2}{c^2} \cdot h\left(\frac{c\varepsilon}{n\sigma^2}\right)\right)$$

mit $h(u) = (1+u)\ln(1+u) - u$ für $u \geq 0$.

Beweis. Mittels Dreiecksungleichung für Integrale und der Linearität des Erwartungswertes folgt für eine reelle Zufallsvariable X mit $k \geq 2$

$$\mathbf{E}\left[X^k\right] \leq \mathbf{E}\left[|X|^k\right] \leq \mathbf{E}\left[|X|^2 c^{k-2}\right] = c^{k-2}\mathbf{E}\left[|X|^2\right] = \sigma^2 c^{k-2}.$$

Aus der Taylorreihe der Exponentialfunktion erhält man

$$\begin{aligned}\mathbf{E}\left[\exp(sX)\right] &= \mathbf{E}\left[1 + sX + \sum_{k=2}^{\infty} \frac{s^k X^k}{k!}\right] \\ &= 1 + \sum_{k=2}^{\infty} \frac{s^k \mathbf{E}\left[X^k\right]}{k!} \\ &\leq 1 + \sum_{k=2}^{\infty} \frac{s^k \sigma^2 c^{k-2}}{k!} \\ &\leq 1 + \frac{\sigma^2}{c^2} \cdot \sum_{k=2}^{\infty} \frac{s^k c^k}{k!} \\ &= 1 + \frac{\sigma^2}{c^2} \cdot (e^{sc} - 1 - sc)\,.\end{aligned}$$

Unter Beachtung von $1 + x \leq \exp(x)$ für $x \geq 0$ folgt dann

$$\leq \exp\left(\frac{\sigma^2}{c^2} \cdot (e^{sc} - 1 - sc)\right).$$

Sei $\sigma_k^2 := \mathbf{Var}[X_k]$. Unter Verwendung des bereits Gezeigten und der Markov'schen Ungleichung ergibt sich mit $s, \varepsilon > 0$

$$\begin{aligned}\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] &= \mathbf{P}\left[\exp\left(s\sum_{k=1}^{n} X_k\right) \geq \exp(s\varepsilon)\right] \\ &\leq \exp(-s\varepsilon)\mathbf{E}\left[\exp\left(s\sum_{k=1}^{n} X_k\right)\right] \\ &= \exp(-s\varepsilon)\mathbf{E}\left[\prod_{k=1}^{n} \exp\left(sX_k\right)\right].\end{aligned} \tag{A.7}$$

Wegen der Unabhängigkeit der X_k sind auch die Zufallsvariablen $\exp(sX_k)$ unabhängig, siehe dazu Satz 4.4.8 in [Beh13, S. 141]. Unter iterativer Anwendung, dass unabhängige Zufallsvariablen unkorelliert sind [Kle13, S. 104], erhält man

$$\mathbf{E}\left[\prod_{k=1}^{j} \exp(sX_k)\right] = \mathbf{E}\big[\exp(sX_j)\big] \cdot \mathbf{E}\left[\prod_{k=1}^{j-1} \exp(sX_k)\right], \qquad \text{für } n \geq j \geq 2.$$

Fasst man diese Gleichung für alle j zusammen, so ergibt sich

$$\mathbf{E}\left[\prod_{k=1}^{n} \exp\left(sX_k\right)\right] = \prod_{k=1}^{n} \mathbf{E}\left[\exp\left(sX_k\right)\right]. \tag{A.8}$$

Kombiniert man nun die Ungleichung (A.7) mit der Gleichung (A.8), so erhält man

$$\begin{aligned}
\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] &\leq \exp(-s\varepsilon) \prod_{k=1}^{n} \mathbf{E}\left[\exp(sX_k)\right] \\
&\leq \exp(-s\varepsilon) \prod_{k=1}^{n} \exp\left(\frac{\sigma_k^2}{c^2}\left[\exp(sc) - 1 - sc\right]\right) \\
&= \exp(-s\varepsilon) \exp\left(\frac{n\sigma^2}{c^2}\left[\exp(sc) - 1 - sc\right]\right) \\
&= \exp\left(\frac{n\sigma^2}{c^2}\left[\exp(sc) - 1 - sc\right] - s\varepsilon\right) \\
&= \exp(f(s)).
\end{aligned} \tag{A.9}$$

Mit

$$f(s) := \frac{n\sigma^2}{c^2}\left[\exp(sc) - 1 - sc\right] - s\varepsilon \qquad \text{und} \qquad f'(s) = \frac{n\sigma^2}{c}(\exp(sc) - 1) - \varepsilon,$$

ergibt sich daraus

$$\begin{aligned}
0 &= \frac{n\sigma^2}{c}(\exp(s_0 c) - 1) - \varepsilon \\
\iff \quad \exp(s_0 c) &= \frac{\varepsilon c}{n\sigma^2} + 1 \\
\iff \quad s_0 &= \frac{1}{c} \cdot \ln\left(\frac{\varepsilon c}{n\sigma^2} + 1\right).
\end{aligned}$$

Damit hat f eine Extremstelle an $s_0 = \frac{1}{c} \cdot \ln\left(\frac{\varepsilon c}{n\sigma^2} + 1\right)$. Bildet man die 2. Ableitung von f und setzt s_0 ein, so folgt

$$f''(s_0) = n\sigma^2 \exp(s_0 c) > 0.$$

Das bedeutet, dass s_0 eine Minimalstelle von f ist. Verwendet man s_0 in der Ungleichung (A.9), so erhält man mit $h(u) = (1+u)\ln(1+u) - u$ dann die Behauptung

$$\begin{aligned}
\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] &\leq \exp\left(\frac{n\sigma^2}{c^2}\left[\exp(s_0 c) - 1 - s_0 c\right] - s_0\varepsilon\right) \\
&= \exp\left(\frac{n\sigma^2}{c^2}\left[\frac{\varepsilon c}{n\sigma^2} - \ln\left(\frac{\varepsilon c}{n\sigma^2} + 1\right)\right] - \frac{\varepsilon}{c}\ln\left[\frac{\varepsilon c}{n\sigma^2} + 1\right]\right) \\
&= \exp\left(\frac{n\sigma^2}{c^2}\left[\frac{\varepsilon c}{n\sigma^2} - \ln\left(\frac{\varepsilon c}{n\sigma^2} + 1\right) - \frac{\varepsilon c}{n\sigma^2}\ln\left(\frac{\varepsilon c}{n\sigma^2} + 1\right)\right]\right) \\
&= \exp\left(-\frac{n\sigma^2}{c^2}\left[\left(1 + \frac{\varepsilon c}{n\sigma^2}\right)\ln\left(1 + \frac{\varepsilon c}{n\sigma^2}\right) - \frac{\varepsilon c}{n\sigma^2}\right]\right) \\
&= \exp\left(-\frac{n\sigma^2}{c^2} \cdot h\left[\frac{\varepsilon c}{n\sigma^2}\right]\right).
\end{aligned}$$

□

Lemma A.15. (Bernstein Ungleichung) *Seien $X_1, \ldots, X_n$ unabhängige reelle Zufallsvariablen mit $\mathbf{E}[X_i] = 0$ und es gelte $|X_i| \leq c$ fast sicher. Sei*

$$\sigma^2 := \frac{1}{n} \sum_{k=1}^{n} \mathbf{Var}[X_k].$$

Dann gilt für jedes $\varepsilon > 0$

$$\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] \leq \exp\left(-\frac{\varepsilon^2}{2n\sigma^2 + 2c\varepsilon/3}\right). \tag{A.10}$$

Beweis. Seien für $u > 0$ folgende Abbildungen definiert:

$$h(u) := (1+u)\ln(1+u) - u \qquad \text{und} \qquad g(u) := \frac{u^2}{2} \cdot \frac{3}{3+u}.$$

Dann ergibt sich

$$h'(u) = \ln(1+u), \qquad g'(u) = \frac{1}{2}\left(\frac{18u + 3u^2}{(3+u)^2}\right),$$
$$h''(u) = \frac{1}{1+u}, \qquad g''(u) = \frac{27}{(u+3)^3}.$$

Es gilt weiter $h(0) = g(0) = h'(0) = g'(0) = 0$ und für $u > 0$ folgt

$$\begin{aligned} & u^3 + 9u^2 > 0 \\ \iff \quad & u^3 + 9u^2 + 27u + 27 > 27u + 27 \\ \iff \quad & (u+3)^3 > 27(u+1) \\ \iff \quad & \frac{1}{u+1} > \frac{27}{(u+3)^3} \\ \iff \quad & h''(u) > g''(u). \end{aligned}$$

Also erhält man

$$h(u) = h(u) - h(0) = \int_0^u h'(s)\,ds = \int_0^u h'(s) - h'(0)\,ds = \int_0^u \int_0^s h''(t)\,dt\,ds$$

$$\leq \int_0^u \int_0^s g''(t)\,dt\,ds = \int_0^u g'(s) - g'(0)\,ds = \int_0^u g'(s)\,ds = g(u) - g(0) = g(u).$$

Aus der Ungleichung von Bennett ergibt sich nun für jedes $\varepsilon > 0$

$$\mathbf{P}\left[\sum_{k=1}^{n} X_k \geq \varepsilon\right] \leq \exp\left(-\frac{n\sigma^2}{c^2} \cdot h\left(\frac{c\varepsilon}{n\sigma^2}\right)\right),$$

da $\exp(-x)$ monoton fallend ist und $h(u) \geq g(u)$ gilt, kann man wie folgt weiter abschätzen

$$\begin{aligned}
&\leq \exp\left(-\frac{n\sigma^2}{c^2} \cdot g\left(\frac{c\varepsilon}{n\sigma^2}\right)\right) \\
&= \exp\left(-\frac{n\sigma^2}{c^2} \cdot \frac{1}{2} \cdot \left(\frac{c\varepsilon}{n\sigma^2}\right)^2 \cdot \frac{3}{3 + \frac{c\varepsilon}{n\sigma^2}}\right) \\
&= \exp\left(-\frac{\varepsilon^2}{2n\sigma^2 + \frac{2c\varepsilon}{3}}\right). \qquad \square
\end{aligned}$$

Die hier verwendeten Beweisideen der Young'schen und Hölder'schen Ungleichung stammen aus [Kle13, S. 152].

Lemma A.16. ***(Young'sche Ungleichung)*** *Sei $p, q \in (1; \infty)$ mit $\frac{1}{p} + \frac{1}{q} = 1$ und $x, y \in [0; \infty)$, dann ist*

$$xy \leq \frac{x^p}{p} + \frac{y^q}{q}.$$

Beweis. Sei $y \in (0, \infty)$ fest gewählt. Betrachte $f(x) := \frac{x^p}{p} + \frac{y^q}{q} - xy$. Es gilt

$$f'(x) = x^{p-1} - y, \qquad f''(x) = (p-1)x^{p-2}.$$

Damit ist $x_0 = y^{1/p-1} > 0$ Extremstelle und $f''(x_0) > 0$ also sogar Minimalstelle. Weiter gilt wegen $q = \frac{p}{p-1}$ und $x_0^p = y^q$

$$f(x_0) = \left(\frac{1}{p} + \frac{1}{q}\right) y^q - y^{p/p-1} = 0.$$

Damit ist $f(x) \geq 0$.
Sei nun $y = 0$. Dann ist trivialerweise $xy = 0 \leq \frac{x^p}{p} = \frac{x^p}{p} + \frac{y^q}{q}$ erfüllt. Also ist die Behauptung gezeigt. $\square$

Lemma A.17. ***(Hölder'sche Ungleichung)*** *Seien $p, q \in [1; \infty]$ mit $\frac{1}{p} + \frac{1}{q} = 1$ und $f \in \mathcal{L}^p(\mu), g \in \mathcal{L}^q(\mu)$. Dann gilt $fg \in \mathcal{L}^1(\mu)$ und*

$$\|fg\|_1 \leq \|f\|_p \cdot \|g\|_q. \tag{A.11}$$

Beweis. Für $p = 1$ gilt

$$\|fg\|_1 = \int |fg| \, d\mu \leq \|g\|_\infty \int |f| \, d\mu = \|g\|_\infty \|f\|_1 < \infty.$$

Analog für $p = \infty$ mit vertauschten Rollen von f und g.
Sei $p \in (1, \infty)$ und $f \in \mathcal{L}^p(\mu)$ und $g \in \mathcal{L}^q(\mu)$ nicht fast überall Null. Dann kann man $f' = \frac{f}{\|f\|_p}$ und $g' = \frac{g}{\|g\|_q}$ setzen und erhält $\|f'\|_p = 1 = \|g'\|_q$. Mit Hilfe der Youngschen Ungleichung folgt dann für $p, q \in (1, \infty)$:

$$\begin{aligned}
\frac{\|fg\|_1}{\|f\|_p \cdot \|g\|_q} &= \|f'g'\|_1 \\
&= \int |f'| \cdot |g|' \, d\mu \\
&\leq \frac{1}{p} \int |f'|^p \, d\mu + \frac{1}{q} \int |g'|^q \, d\mu \\
&= \frac{1}{p} + \frac{1}{q} = 1 = \|f'\|_p \|g'\|_q = \frac{\|f\|_p \|g\|_q}{\|f\|_p \cdot \|g\|_q}. \\
\Leftrightarrow \qquad \|fg\|_1 &\leq \|f\|_p \|g\|_q .
\end{aligned}$$

□

Das folgende Lemma mit Beweis findet man in [Kle13, S. 100].

Lemma A.18. *Sei $(\Omega, \mathfrak{A}, \mu)$ ein Maßraum und $f \colon \Omega \to \mathbb{R}$ messbar, sowie $f \geq 0$ fast überall. Dann gilt*

$$\int f \, d\mu = \int_0^\infty \mu(\{f \geq t\}) \, \mathrm{d}t. \tag{A.12}$$

Die nächste Definition stammt aus [Dei14, S. 138].

Definition A.19. Seien $(a_n)_{n\in\mathbb{N}}$ und $(b_n)_{n\in\mathbb{N}}$ zwei Folgen reeller Zahlen mit $a_n \neq 0 \neq b_n$ für alle n. Die Folgen heißen asymptotisch gleich, geschrieben $a_n \sim b_n$, falls

$$\lim_{n\to\infty} \frac{a_n}{b_n} = 1.$$

Vergleiche mit [Hil06, S. 20] für die nachfolgende Definition.

Definition A.20. Wir definieren $\lfloor x \rfloor := \max\{k \in \mathbb{Z} \mid k \leq x\}$ und $\lceil x \rceil := \min\{k \in \mathbb{Z} \mid k \geq x\}$ als die Abrundungs- bzw. Aufrundungsfunktion.

Lemma A.21. *Sei $s \in (0;\infty)$ und $u \in (0{,}5;1)$. Dann nimmt $f\colon (0;\infty) \to (0;\infty)$ definiert durch*

$$f(m) := \left(\frac{m \ln n}{n}\right)^u + m^{-s}$$

für $n \in \mathbb{N}$ ein lokales Minimum an der Stelle

$$m_0 := \left(\frac{s}{u}\right)^{\frac{1}{s+u}} \cdot \left(\frac{n}{\ln n}\right)^{\frac{u}{u+s}}$$

an. Außerdem gilt für $k \in \{1,\dots,n\}$ mit $k > \lceil m_0 \rceil$ dann

$$\min_{m\in\{1,\dots,k\}} f(m) = \min\{f(\lceil m_0 \rceil), f(\lfloor m_0 \rfloor)\} \sim f(m_0).$$

Beweis. Für die Ableitungen von f gilt

$$f'(m) = um^{u-1}\left(\frac{\ln n}{n}\right)^u - sm^{-s-1}$$

und

$$f''(m) = u(u-1)m^{u-2}\left(\frac{\ln n}{n}\right)^u + s(s+1)m^{-s-2}.$$

Setzt man f' gleich Null, so erhält man

$$\begin{aligned}
& 0 = um_0^{u-1}\left(\frac{\ln n}{n}\right)^u - sm_0^{-s-1} \\
\Leftrightarrow \quad & m_0^{u-1+s+1} = \frac{s}{u}\left(\frac{n}{\ln n}\right)^u \\
\Leftrightarrow \quad & m_0 = \left(\frac{s}{u}\right)^{\frac{1}{s+u}} \cdot \left(\frac{n}{\ln n}\right)^{\frac{u}{u+s}}.
\end{aligned}$$

Als kleine Vorbetrachtung für das Einsetzen von m_0 in die zweite Ableitung von f, zeigen wir, dass mit dem Ansatz $s = au$ für $a > 0$ die folgende Ungleichung gilt:

$$u(u-1)\left(\frac{s}{u}\right)^{\frac{u}{u+s}} + s(s+1)\left(\frac{u}{s}\right)^{\frac{s}{s+u}} > 0.$$

Das erreicht man wie folgt:

$$\begin{aligned}
& u(u-1)\left(\frac{s}{u}\right)^{\frac{u}{u+s}} + s(s+1)\left(\frac{u}{s}\right)^{\frac{s}{s+u}} \\
&= u(u-1)a^{\frac{1}{1+a}} + au(au+1)a^{-\frac{a}{a+1}} \\
&= u^2a^{\frac{1}{1+a}} - ua^{\frac{1}{1+a}} + a^2u^2a^{-\frac{a}{a+1}} + aua^{-\frac{a}{a+1}} \\
&= u^2a^{\frac{1}{1+a}} + a^2u^2a^{-\frac{a}{a+1}} > 0.
\end{aligned}$$

Nun zeigt man noch den Typ des Extremums:

$$\begin{aligned}
f''(m_0) &= u(u-1)m_0^{u-2}\left(\frac{\ln n}{n}\right)^u + s(s+1)m_0^{-s-2} \\
&= u(u-1)\left(\frac{s}{u}\right)^{\frac{u-2}{s+u}} \underbrace{\left(\frac{n}{\ln n}\right)^{\frac{(u-2)u}{u+s}} \left(\frac{\ln n}{n}\right)^u}_{=\left(\frac{n}{\ln n}\right)^{\frac{u^2-2u-u^2-us}{u+s}}=\left(\frac{n}{\ln n}\right)^{-\frac{u(s+2)}{u+s}}} + s(s+1)\left(\frac{n}{\ln n}\right)^{-\frac{u(s+2)}{u+s}}\left(\frac{s}{u}\right)^{-\frac{(s+2)}{s+u}} \\
&= \underbrace{\left(\frac{n}{\ln n}\right)^{-\frac{u(s+2)}{u+s}}}_{>0} \underbrace{\left(\frac{s}{u}\right)^{-\frac{2}{s+u}}}_{>0} \underbrace{\left[u(u-1)\left(\frac{s}{u}\right)^{\frac{u}{u+s}} + s(s+1)\left(\frac{u}{s}\right)^{\frac{s}{s+u}}\right]}_{>0} > 0.
\end{aligned}$$

Somit ist m_0 eine Minimalstelle von f.

Da f stetig ist und keine weitere lokale Extremstelle besitzt, erfolgt der einzige Monotoniewechsel bei der Minimalstelle m_0 von monoton fallend zu monoton wachsend. Somit gilt für $l, l' \in \mathbb{N}$ mit $l < l' \leq \lfloor m_0 \rfloor$:

$$f(l) \geq f(l') \geq f(\lfloor m_0 \rfloor) \geq f(m_0),$$

da f monoton fallend für $m < m_0$ ist. Analog folgt für $l, l' \in \mathbb{N}$ mit $\lceil m_0 \rceil \leq l < l'$:

$$f(m_0) \leq f(\lceil m_0 \rceil) \leq f(l) \leq f(l'),$$

weil f monoton wachsend für $m > m_0$ ist. Insgesamt erhält man für $k \in \{1, \ldots, n\}$ mit $k > \lceil m_0 \rceil$ dann

$$\min_{m \in \{1,\ldots,k\}} f(m) = \min\{f(\lceil m_0 \rceil), f(\lfloor m_0 \rfloor)\}.$$

Es gilt

$$m_0 = \left(\frac{s}{u}\right)^{\frac{1}{s+u}} \cdot \left(\frac{n}{\ln n}\right)^{\frac{u}{u+s}} \longrightarrow \infty,$$

für $n \to \infty$. Sei $\lambda \in (-1; 1)$. Dann ist

$$\begin{aligned}
\frac{f(m_0)}{f(m_0+\lambda)} &= \frac{m_0^u \cdot \left(\frac{\ln n}{n}\right)^u + m_0^{-s}}{(m_0+\lambda)^u \cdot \left(\frac{\ln n}{n}\right)^u + (m_0+\lambda)^{-s}} \\
&= \frac{\left(\frac{\ln n}{n}\right)^u + m_0^{-s-u}}{\left(\frac{m_0+\lambda}{m_0}\right)^u \cdot \left(\frac{\ln n}{n}\right)^u + m_0^{-u-s}\left(\frac{m_0}{m_0+\lambda}\right)^s} \\
&= \frac{\left(\frac{\ln n}{n}\right)^u + \frac{u}{s} \cdot \left(\frac{\ln n}{n}\right)^u}{\left(\frac{m_0+\lambda}{m_0}\right)^u \cdot \left(\frac{\ln n}{n}\right)^u + \frac{u}{s} \cdot \left(\frac{\ln n}{n}\right)^u \left(\frac{m_0}{m_0+\lambda}\right)^s}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1+\frac{u}{s}}{\left(1+\frac{\lambda}{m_0}\right)^u + \frac{u}{s}\cdot\left(\frac{1}{1+\frac{\lambda}{m_0}}\right)^s} \\
&\longrightarrow \frac{1+\frac{u}{s}}{(1+0)^u + \frac{u}{s}\cdot\left(\frac{1}{1+0}\right)^s} \\
&= \frac{1+\frac{u}{s}}{1+\frac{u}{s}} = 1
\end{aligned}
$$

für $n \to \infty$. Als einfache Folgerung aus Definition A.20 erhält man die Existenz eines $\lambda_1 \in (-1;0]$ und eines $\lambda_2 \in [0;1)$ mit

$$\lfloor m_0 \rfloor = m_0 + \lambda_1, \qquad \lceil m_0 \rceil = m_0 + \lambda_2.$$

Also ist auch

$$\lim_{n\to\infty} \frac{f(m_0)}{f(\lceil m_0 \rceil)} = 1 = \lim_{n\to\infty} \frac{f(m_0)}{f(\lfloor m_0 \rfloor)}.$$

Somit ist per Definition A.19 der asymptotischen Gleichheit dann

$$f(m_0) \sim f(\lfloor m_0 \rfloor), \qquad f(m_0) \sim f(\lceil m_0 \rceil)$$

und der Transitivität dieser Relation auch

$$f(m_0) \sim f(\lfloor m_0 \rfloor) \sim f(\lceil m_0 \rceil) \sim \min\{f(\lceil m_0 \rceil), f(\lfloor m_0 \rfloor)\}$$

erfüllt. □

B | Kenngrößen

Hier wird dargestellt, wie sich die Kenngrößen für Klassifikatoren aus den Kontingenztafeln ergeben, vgl. dazu [Kuh+16, S. 24 f.] und [Jam+16, S. 148 f.].

Vorhersage \ Referenz	1	2
1	TP	FP
2	FN	TN

Tabelle B.1: Kontingenztafel zur Diagnoseprüfung

TP steht dabei für richtig positiv, FP für falsch positiv, FN für falsch negativ und TN für richtig negativ. Diese Bezeichnungen stammen aus der Epidemiologie bei der Diagnoseprüfung einer Krankheit. Dabei lässt sich die Erkennung einer Krankheit als positiv interpretieren und das Gegenteil als negativ. Hier stellt die Klasse 1 den positiven und Klasse 2 den negativen Fall dar.

Kenngröße		Formel
Treffergenauigkeit	=	(TP + TN)/(TP+FP+FN+TN)
Sensitivität	=	TP/(TP+FN)
Spezifität	=	TN/(FP+TN)
Prävalenz	=	(TP+FN)/(TP+FP+FN+TN)
Positiver Vorhersagewert	=	TP/(TP+FP)
Negativer Vorhersagewert	=	FN/(FN+TN)
Erkennungsrate	=	TP/(TP+FP+FN+TN)
Erkannte Prävalenz	=	(TP+FP)/(TP+FP+FN+TN)
Ausbalancierte Genauigkeit	=	(Sensitivität+Spezifität)/2

Tabelle B.2: Kenngrößentabelle mit Formeln

C | Verwendete Hard- und Software

C.1 R

R: 3.2.3
Rcpp: 0.12.3
caret: 6.0.64
tree: 1.0.37
e1071: 1.6.7
MASS: 7.3.45
nnet: 7.3.11
randomForest: 4.6.12
class: 7.3.14
roxygen2: 5.0.1
testthat: 0.11.0
knitr: 1.12.3
ggplot2: 2.0.0
tikzDevice: 0.10.1
xtable: 1.8.2
ODT: 0.1
CUDTP: 0.1
SLF: 0.1

C.2 Anderes

Das Anlegen der Projekte erfolgte mit der RStudio IDE. Bei der Kompilierung der Algorithmen CUDTP und ODT wurde C++11 verwendet und für CUDTP kam zusätzlich die Bibliothek GMP der Version 6.1.0. zum Einsatz. Zur Erstellung der Abbildungen wurden neben den R internen bzw. durch Pakete bereitgestellten Hilfsmitteln die Sprachen PGF und TikZ der Version 3.0.1a benutzt.
Die Versuche wurden mit dem Betriebssystem Arch Linux x86_64 der Kernelversion 4.4.1-2 durchgeführt. Dabei stand ein Intel Core i3-370M Prozessor, sowie 3629MB RAM zur Verfügung.

Quelltext: Hier die verschiedenen Programme

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche und sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann.

Ort Datum Unterschrift