HUMBOLDT-UNIVERSITÄT ZU BERLIN

Wirtschaftswissenschaftliche Fakultät

BACHELORARBEIT

Bachelor (B.Sc.) in Volkswirtschaftslehre

Vergleich von Vorhersagemodellen zu Stornierungen von Hotelbuchungen

Comparison of predictive models on hotel booking cancellations

Julius Freidank

Matrikelnr.: 588699

Erstgutachterin: Prof. Dr. Sonja Greven
Zweitgutachterin: Prof. Dr. Nadja Klein
Betreuer: Dr. Sigbert Klinke
Berlin, den 09.08.2021

Inhaltsverzeichnis

1 Einleitung

Das Stornieren von bestehenden Hotelbuchungen durch Kunden stellt für die Hotelindustrie ein Problem dar. Wenn diese kurz vor Beginn des Buchungszeitraums auftreten, können stornierte Zimmerbuchungen nicht ohne Weiteres direkt erneut vergeben werden, da Reisebuchungen oft nach langfristiger Planung seitens der Kunden vollzogen werden.

Hotels haben Anreize, mögliche Ausfälle zu senken, um Kosten zu minimieren und Gewinne zu maximieren. Dafür werden verschiedene Ansätze verwendet, um die Zimmer erneut zu vergeben. Ein Ansatz ist, dass die Hotels *Last-Minute*-Angebote veröffentlichen. Dabei werden meistens Reisen angeboten, die wenige Tage nach Veröffentlichung des Angebots stattfinden. Damit potenzielle Kunden diese wählen, sind die Angebote im Preis reduziert. Die Vorteile dieser Methode sind, dass stornierte Zimmer vergeben und Kosten gesenkt werden können. Außerdem kann der niedrige Preis dazu beitragen, weitere Stammkunden zu gewinnen, weil Kunden sich womöglich das Hotel aufgrund des niedrigeren Preises merken und deswegen erneut Zimmer buchen.

Eine weitere gängige, jedoch kontroversere Methode ist das Überbuchen von Hotelzimmern durch Hotels. Überbuchungen sind das Anbieten von mehr Hotelzimmern, als es die Kapazitäten tatsächlich zulassen. Mögliche Ausfälle von Buchungen können damit durch einen Überschuss an Kundenbuchungen ausgeglichen werden. Diese Methode erlaubt den Hotels eine bessere Planbarkeit der zukünftigen Umsätze, da bei einem Buchungsausfall das Hotelzimmer direkt auf eine Überbuchung umgelegt werden kann. Das bedeutet also auch, dass die Gewinne höher ausfallen als bei Sonderangeboten. Nachteil dieser Praxis ist unter anderem der Verlust der Reputation aufseiten des Hotels, da durch das Überbuchen einige Kunden bei hoher Auslastung des Hotels vor Reiseantritt eine Absage erhalten oder einem anderen Hotel zugewiesen werden (vgl. Toh et al. 2005, S. 125). Kunden könnten aus diesem Grund den Kundenstamm verlassen.

Bisher wird die Anzahl der überbuchten Zimmer beispielsweise nach der Rate der Stornierungen der Vorjahre ermittelt (vgl. Toh et al. 2005, S. 125). Modelle, die die Stornierungen vorhersagen, könnten eine zielgerichtetere Überbuchung der Hotelzimmer erlauben. Um das Potenzial dieser Modelle zu überprüfen, werden in dieser Arbeit verschiedene Methoden genutzt, die es ermöglichen sollen, Modelle zu erstellen, die Stornierungen von Hotelbuchungen anhand einiger bestimmter Merkmale vorherzusagen. Dabei werden verschiedene Methoden des überwachten maschinellen Lernens verwendet. Als Teilbereich der Informatik ist das Ziel von maschinellem Lernen mithilfe von Algorithmen Muster und Zusammenhänge zu erkennen und daraus Lösungen für Probleme zu entwickeln (vgl. Rebala et al. 2019, S. 1-2).

Das Ziel ist es, die Ergebnisse verschiedener Modelle miteinander zu vergleichen und zu evaluieren, ob und wie gut maschinelles Lernen geeignet ist, Stornierungen vorherzusagen. Romero Morales et al. (2010) behaupten, dass es schwer möglich wäre die Stornierungen

von Hotelbuchungen anhand der Buchungsdaten vorherzusagen und wenn dann wäre dies nur mit einer niedrigen Genauigkeit möglich. Jedoch widerlegen Antonio, De Almeida et al. (2017) diese Behauptung unter anderem durch die Verwendung von künstlichen neuronalen Netzen, die ebenfalls in dieser Arbeit genutzt werden. Dabei erreichen sie mit ihren Modellen größtenteils Vorhersagegenauigkeiten von über 90 % und teilweise Genauigkeiten von beinahe 99 %.

2 Datensatz

Der in dieser Arbeit verwendete Datensatz wurde von *Kaggle.com* entnommen und stammt ursprünglich aus einem wissenschaftlichen Artikel von Antonio, de Almeida et al. (2019) (vgl. Mostipak 2020). Die Daten beschreiben dabei ein Resort-Hotel und ein Stadt-Hotel in Portugal. Der Unterschied zwischen diesen beiden Hotelarten besteht darin, dass Ersteres in einer Hotelanlage liegt und abseits des Hotelgeschäfts weitere Annehmlichkeiten wie etwa Freizeit- und Sportangebote und oftmals mehrere Restaurants anbietet. Damit halten sich Gäste während des Buchungszeitraums hauptsächlich im Resort auf. Stadt-Hotels hingegen liegen im innerstädtischen Raum ohne größere Hotelanlage und dienen hauptsächlich der Beherbergung von Gästen.

Der Datensatz umfasst insgesamt 119.390 Beobachtungen. Die relevanten Variablen werden in Tabelle 2.1 dargestellt und beschrieben. Der Satz beinhaltet Daten vom 1. Juli 2015 bis zum 31. August 2017 (vgl. zu diesem Abschnitt Antonio, de Almeida et al. 2019, S. 41). Es werden 31 Variablen bereitgestellt, jedoch werden nicht alle Variablen für die nachfolgende Analyse genutzt, da sie teilweise redundante Informationen liefern und aufgrund einiger Eigenschaften die Analyse erschweren. Es werden nur 18 Variablen in die Analyse übernommen. Die Variable über die Aufenthaltsdauer wird aus zwei Variablen erstellt. Eine gab die Anzahl der Aufenthalte über Wochenenden an und die andere die Anzahl der Aufenthalte unter der Woche an.

Tabelle 2.1: Beschreibungen und Ausprägungen der verwendeten Variablen

Variablenname	Beschreibung und Ausprägungen
ist_storn	Stornierungsstatus; 1 = storniert, 0 = nicht storniert
hotel	Art des Hotels; Stadthotel oder Resort
vorlauf	Vergangene Tage zwischen Buchung und Reiseantritt
erwachs	Anzahl der Erwachsenen
kinder	Anzahl der Kinder
klnkndr	Anzahl der Kleinkinder
mahl	Art der gebuchten Mahlzeiten; FP = Frühstückspension, VP= Vollpension, HP= Halbpension, KM= keine Mahlzeit
marktsg	Zugewiesenes Marktsegment der Buchung; Luftfahrt = Räume für Flugzeugbesatzung, Komplementär = sonstige, Unternehmen = Buchungen durch Unternehmen, Direkt = Buchungen direkt beim Hotel, Gruppen = Gruppenbuchungen, Reiseveranstalter = Buchungen bei Reiseveranstaltern, Online = Buchungen über Onlineportale
vtrb	Vetriebsweg der Buchung; Unternehmen = Buchungen durch Unternehmen, Direkt = Buchung bei Hotel, GDS = Buchung über Reservierungssystem, RB/RV = Buchung bei Reisebüro oder Reiseveranstalter
wdh_gast	Gibt an, ob der Hotelgast schon einmal das Hotel gebucht hat; 1 = erneute Buchung, 0 = neuer Gast
vorher_storn	Anzahl der vorherigen Stornierungen beim Hotel
vorher_n_storn	Anzahl vorheriger Buchungen, die nicht storniert wurden
rsrv_rm	Art des Hotelzimmers; Kodierte Werte von A bis P
anzahlen	Art der Anzahlung; Keine = keine Anzahlung wurde geleistet, NichtErstattbar = Anzahlung in Höhe der gesamten Kosten wurde geleistet und ist nicht erstattbar, Erstattbar = Anteil der gesamten Kosten wurde angezahlt
liste	Anzahl der Tage in der Warteliste, bevor Buchung vollzogen werden konnte
kunde	Buchungsart; Vertrag = Buchung ist mit Vertrag verbunden, Gruppe = Buchung ist mit Gruppe verbunden, Kurzfristig = weder mit Vertrag und Gruppe und nicht mit anderer Buchung verbunden, Kurzfristig_P = Buchung ist Kurzfristig und mit anderer Buchung verbunden
dtr	Durchschnittliche tägliche Kosten des Hotelzimmers
parken	Anzahl der benötigten Parkplätze vom Gast
dauer	Aufenthaltsdauer der Kunden im Hotel; in Nächte

Quelle: Antonio, de Almeida et al. (2019, S. 43-44)

Anschließend wird der Datensatz in drei verschiedene Datensätze aufgeteilt. 60 % der Daten werden dabei einem Trainingsdatensatz, 20 % einem Validierungsdatensatz und die restlichen 20 % einem Testdatensatz zugewiesen. Auf Grundlage des Trainingsdatensatzes werden die Vorhersagemodelle erstellt und dabei werden verschiedene Entscheidungen bezüglich Modellaufbau und der Parameterwahl getroffen. Die verschiedenen erstellten Modelle werden daraufhin mit dem Validierungssatz überprüft. Dabei findet auch die Wahl der optimalen Parameter statt. Die Vorhersageleistungen der verschiedenen Modelle, basierend auf verschiedenen Parametern, werden auf Basis des Validierungsdatensatzes miteinander verglichen und es werden womöglich noch kleinere Änderungen an den Modellparametern vorgenommen, um das bestmögliche Modell zu erhalten. Am Ende dieser Phase werden die finalen Modelle der verschiedenen Methoden genutzt, um die Vorhersageleistung bei einem bisher ungesehenen Testdatensatz zu bestimmen. Durch die Nutzung der verschiedenen Datensätze soll eine Überanpassung verhindert werden. Die Modelle könnten sich ansonsten an besondere Eigenschaften in den Daten des Datensatzes, die normalerweise nicht auftreten, anpassen und so eine schlechtere Leistung erbringen (vgl. zu diesem Abschnitt Aggarwal 2018, S. 178-179).

Der verwendete Datensatz ist unausgewogen, dass bedeutet, dass sich die Anzahl der Buchungen mit Buchungsstornierung erheblich von der Anzahl der Buchungen ohne Stornierung unterscheidet. Etwa 62,96 % der Buchungen wird nicht storniert, während 37,04 % storniert werden.

2.1 Datenaufbereitung

Vor Beginn einer jeden Datenanalyse ist es ratsam, die Daten aufzubereiten. Fehlende Werte, Ausreißer und unmögliche Werte können Analysen verzerren oder sogar komplett verfälschen.

2.1.1 Fehlende Werte

Fehlende Werte treten bei Variablen auf, wenn diese keine Einträge haben. Die Gründe für das Auftreten können verschieden sein. Hierbei unterteilt man das Fehlen in drei verschiedene Kategorien. Die erste Kategorie ist *Missing completely at random* (MCAR). Diese bezeichnet die fehlenden Werte, die zufällig auftreten und unabhängig von der Beobachtung sind. Sie treten dabei also nicht systematisch auf. Fehlende Werte dieser Art sind ungewöhnlich und werden in den meisten Fällen ignoriert beziehungsweise entfernt.

Die nächste Kategorie ist *Missing at random* (MAR). Die fehlenden Werte treten nicht komplett zufällig auf. So können sie beispielsweise nur bei einer bestimmten Variablen auftreten, aber dort unregelmäßig.

Die letzte Art der fehlenden Werte sind jene, die nicht zufällig auftreten. Die dazugehörige Kategorie wird *Missing not at random* (MNAR) genannt. Diese hängen oftmals mit den Werten der Variablen zusammen und treten unter anderem bei Umfragen auf, wenn Befragte sich dazu entscheiden, sensible Fragen nicht zu beantworten.

Insgesamt fehlen nur bei vier Beobachtungen die Werte. Bei jeder einzelnen Variable dieser Beobachtungen fehlen die Einträge. Die Gründe für das Auftreten dieser fehlenden Werte können verschieden sein. Da die gesamten Variablenwerte fehlen, kann man davon ausgehen, dass diese Einträge fehlerhaft sind. Diese Fehler könnten beispielsweise beim Auslesen der Daten auftreten. Die genaue Ursache lässt sich an dieser Stelle jedoch nicht ohne Weiteres ermitteln.

Da die fehlenden Werte dieses Datensatzes willkürlich auftreten und weil weitere Daten über Ursachen fehlen, werden die zugehörigen Beobachtungen von der weiteren Analyse ausgeschlossen. Außerdem sollte das Entfernen von vier Beobachtungen bei über 110.000 Beobachtungen kaum bis gar keine relevanten Auswirkungen auf die Modelle haben.

2.1.2 Ausreißer und ungewöhnliche Werte

Als Nächstes wird der Datensatz auf Ausreißer und ungewöhnliche Werte überprüft. Einige der Variablen haben bedenkliche oder fragwürdige Werte, da diese weit von der Masse der anderen Werte entfernt sind. So gibt es einige Buchungen mit mehr als 50 Erwachsenen oder gar mit keinen Erwachsenen. Außerdem gibt es Zimmer, die 5400 € pro Tag kosten.

Diese Werte mögen zwar fragwürdig sein und könnten fehlerhaft sein, jedoch sind sie nicht gänzlich unmöglich. Hotelbuchungen mit mehr als 50 Erwachsenen könnten Unternehmensreisen oder lediglich große Reisegruppen sein. Reisen ohne Erwachsene könnten von Jugendlichen nahe der Volljährigkeit gebucht worden sein. Hohe Zimmerpreise können mit einer Luxusausstattung zu erklären sein. Aufgrund dessen, dass der Datensatz auf realen Daten basiert und dass diese Werte bei realen Buchungen ebenfalls auftreten können, wird sich an dieser Stelle dazu entschieden, keine der ungewöhnlichen Werte auszuschließen. Ziel ist es auch, dass die Vorhersagemodelle mit diesen Werten zuverlässige Vorhersagen treffen und so auf ungewöhnliche Werte reagieren können.

3 Methoden

3.1 Multiple logistische Regressionen

3.1.1 Einführung

Logistische Regressionsmodelle ähneln vom Prinzip den linearen Regressionsmodellen. Ziel ist es auch hier unter Verwendung von abhängigen Variablen eine unabhängige

Variable zu beschreiben, Beziehungen interpretierbar zu machen und Informationen über die Stärke der Abhängigkeiten zu gewinnen. Während bei der linearen Regression die abhängige Variable metrischskaliert ist, ist sie bei der logistischen Regression binomial oder dichotom (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 1).

Das Ziel der logistischen Regression ist es, die bedingte Wahrscheinlichkeit $E(Y|X = x)$ einer Variablen Y zu ermitteln, also der erwartete Wert der abhängigen Variable Y gegeben dem Wert x der abhängigen Variablen X. Bei einer linearen Regression kann $E(Y|x)$ jeden Wert von $-\infty$ und $+\infty$ annehmen. Wenn der Regressand, also die abhängige Variable, jedoch dichotom ist, nimmt der Erwartungswert Werte von $(0 \leq E(Y|x) \leq 1)$ an. Das liegt daran, dass die Variable nur Werte von 0 und 1 annehmen kann und der Erwartungswert dann Wahrscheinlichkeiten für das Eintreten eines Ereignisses angibt. Das multiple logistische Regressionsmodell lässt sich wie folgt beschreiben (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 2-7):

$$E(Y|x) = \pi(x) = \frac{e^{(\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_p x_p)}}{1 + e^{(\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_p x_p)}} \tag{3.1}$$

In (3.1) entspricht $\pi(x)$ der Wahrscheinlichkeit eines Eintretens eines Ereignisses und x_p einer unabhängigen Variablen an der Stelle p. Der Regressionskoeffizient β_p ist der Variablen x_p zugehörig. Ein wichtiger Bestandteil von logistischen Regressionen ist die Logit-Transformation. Sie ist namensgebend für Logit-Modelle, einem weiteren Ausdruck für logistische Regressionsmodelle. Logit bezeichnet dabei den natürlichen Logarithmus einer Chance. Die Transformation mit Fehlerterm ε wird folgendermaßen definiert:

$$g(x) = ln\left[\frac{\pi(x)}{1-\pi(x)}\right] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \varepsilon \tag{3.2}$$

Die Transformation hat etliche Vorteile gegenüber der ursprünglichen Funktion. Wie in (3.2) zu sehen ist, ähnelt die rechte Seite einer linearen Regression. Das bedeutet, dass $g(x)$ stetig sein kann, linear in den Parametern ist und Werte von $-\infty$ und $+\infty$ annehmen kann (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 7).

Nach Hosmer Jr et al. (2013) kann bei der linearen Regression die abhängige Variable als $y = E(Y|x) + \varepsilon$ definiert werden. ε entspricht dabei dem Fehlerterm, der all jene Einflüsse abbildet, die nicht durch die unabhängigen Variablen abgebildet werden. Diese Einflüsse können strukturell oder zufällig auftreten. Die Fehlerterme bei einer linearen Regression sind normalverteilt, durchschnittlich null und sind konstant. Analog zum Wert der abhängigen Variable bei der linearen Regression kann die abhängige Variable bei einer logistischen Regression mit $y = \pi(x) + \varepsilon$ beschrieben werden. Da der Regressand nur Werte von 0 und 1 annehmen kann, nimmt ε ebenfalls nur zwei Werte an. $\varepsilon = -\pi(x)$, wenn y den Wert 0 annimmt und $\varepsilon = 1-\pi(x)$, wenn y den Wert 1 annimmt. Die abhängige

Variable folgt dann bei einer logistischen Regression einer Binomialverteilung mit einer Wahrscheinlichkeit $\pi(x)$, die dem bedingten Erwartungswert $E[Y|x]$ entspricht (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 7).

3.1.2 Anpassen der logistischen Regression

Um das logistische Regressionsmodell in der Gleichung (3.1) an einen Datensatz anzupassen, müssen die unbekannten Parameter $(\beta_0, \beta_1,...,\beta_p)$ geschätzt werden. Bei einer linearen Regression wird dabei zum Beispiel die Methode der kleinsten Quadrate angewendet. Nach dieser werden die Werte der Parameter so gewählt, dass die Summe der quadratischen Abweichungen minimiert werden. Nach üblichen Annahmen liefert diese Methode dann eine Reihe von gewünschten Eigenschaften. Aufgrund dessen, dass der Regressand im Fall der logistischen Regression dichotom ist, haben die Schätzer nicht diese Eigenschaften (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 8).

Eine oft verwendete Schätzmethode zur Erstellung von Schätzfunktionen ist die Maximum-Likelihood-Methode. Ziel der Methode ist es, Werte zu ermitteln, die die Wahrscheinlichkeit maximieren, die beobachteten Daten zu enthalten. Diese Werte werden Maximum-Likelihood-Schätzer genannt. Für ein logistisches Regressionsmodell mit dichotomer abhängiger Variable können die Werte wie folgt gefunden werden (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 8).

$\pi(x)$ aus (3.1) liefert die bedingte Wahrscheinlichkeit, dass Y gleich 1 ist, gegeben x. Daraus folgt, dass $1 - \pi(x)$ die bedingte Wahrscheinlichkeit, dass Y gleich 0 gegeben x ist (Pr$(Y = 0|x)$). Das bedeutet, dass für die Paare (x_i,y_i) bei $y_i = 1$ der Beitrag zur Likelihood-Funktion $\pi(x_i)$ ist und bei $y_i = 0$ ist dieser Beitrag $1 - \pi(x_i)$. Dabei ist $\pi(x_i)$ die bedingte Wahrscheinlichkeit an der Stelle x_i. Der Beitrag kann durch (3.3) ausgedrückt werden (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 8).

$$\pi(x_i)^{y_i}[1 - \pi(x_i)]^{1-y_i} \tag{3.3}$$

Da die Beobachtungen unabhängig voneinander sind, kann die Likelihood-Funktion als Produkt des Ausdrucks in (3.3) beschrieben werden, wobei $\boldsymbol{\beta} = (\beta_0,\beta_1,...,\beta_p)$ dem Vektor der Parameter entspricht und n die Anzahl der Beobachtungen sei.

$$l(\boldsymbol{\beta}) = \prod_{i=1}^{n} \pi(x_i)^{y_i}[1 - \pi(x_i)]^{1-y_i} \tag{3.4}$$

Um das Arbeiten mit der Likelihood-Funktion zu erleichtern, wird diese für gewöhnlich logarithmiert. Nach der Log-Transformation erhält man:

$$L(\boldsymbol{\beta}) = ln[l(\beta)] = \sum_{i=1}^{n} \{y_i ln[\pi(x_i)] + (1 - y_i) ln[1 - \pi(x_i)]\} \tag{3.5}$$

Durch das Maximieren dieser Funktion erhält man die Maximum-Likelihood-Schätzer. Die Gleichungen, die aus dem Maximieren, also aus dem Ableiten nach β_0, β_1, bis β_p, entstehen, lauten (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 9):

$$\sum_{i=1}^{n} [y_i - \pi(x_i)] \tag{3.6}$$

$$\sum_{i=1}^{n} x_i [y_i - \pi(x_i)] \tag{3.7}$$

3.1.3 Interpretation der Koeffizienten logistischer Regressionsmodelle

Während die Interpretation linearer Regressionsmodelle intuitiv und recht simpel ist, ist die Interpretation logistischer Modelle komplizierter. Um die Koeffizienten einer logistischen Regression zu verstehen, ist es nötig, ein Verständnis für Chancenverhältnisse (oder auch Quotenverhältnisse) zu entwickeln. Im Englischen werden diese *Odds Ratios* (OR) genannt. Quotenverhältnisse geben bei dichotomen Variablen die Chance des Eintreffens eines Ereignisses im Verhältnis zur Chance des Eintretens eines Gegenereignisses an.

Für die Berechnung eines Quotenverhältnisses einer dichotomen unabhängigen Variable ist folgende Gleichung von Nöten.

$$OR = \frac{\dfrac{\pi(1)}{1 - \pi(1)}}{\dfrac{\pi(0)}{1 - \pi(0)}} \tag{3.8}$$

Wie in (3.2) beschrieben, wird die Chance als Quotient einer Wahrscheinlichkeit eines Eintreffens eines Ereignisses und der Wahrscheinlichkeit eines Eintreffens eines Gegenereignisses definiert. Der obere Teil der Gleichung in (3.8) beschreibt die Chance, dass $x = 1$ ist und der untere Teil, dass $x = 0$ ist. Mithilfe von (3.1) lässt sich daraus die Gleichung

$$OR = \frac{\dfrac{\left(\dfrac{e^{\beta_0 + \beta_1}}{1 + e^{\beta_0 + \beta_1}}\right)}{\left(\dfrac{1}{1 + e^{\beta_0 + \beta_1}}\right)}}{\dfrac{\left(\dfrac{e^{\beta_0}}{1 + e^{\beta_0}}\right)}{\left(\dfrac{1}{1 + e^{\beta_0}}\right)}} = e^{\beta_1} \tag{3.9}$$

erstellen. Daraus folgt, dass sich der Regressionskoeffizient aus dem logarithmierten Quotenverhältnis ergibt (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 50-51). Analog dazu lässt sich für polychotome unabhängige Variablen, also nominalskalierten Variablen mit mehr als zwei Ausprägungen, der Regressionskoeffizient ebenfalls anhand der Chancenverhältnisse berechnen. Jedoch ergibt sich das Verhältnis hier aus der Chance eines Eintretens eines Ereignisses und der Chance eines Eintretens eines bestimmten Referenzereignisses. Das Verhältnis wird also immer im Bezug auf einen Referenzwert angegeben (vgl. zu diesem Abschnitt Hosmer Jr et al. 2013, S. 56-57).

3.2 Random Forests

Random Forests wurden von Breiman (2001) entwickelt, der damit auf seine Idee der Regressions- und Klassifikationsbäume aufbaute. Breiman (2001) definiert Random Forests als Klassifikator einer Sammlung von Klassifikatoren, die nach Bäumen strukturiert sind. Damit sind Random Forests Ensembleverfahren mit Regressions- beziehungsweise mit Klassifikationsbäumen. Daher auch Forest, da mehrere Bäume erstellt werden.

Für die Erstellung von Random Forests werden Zufallsvektoren generiert, die das Wachstum jedes Entscheidungsbaums kontrollieren (vgl. Breiman 2001, S. 5). Dabei wird das sogenannte *Bagging* genutzt, welches von Breiman (1996) vorgestellt wurde. Der Begriff ist ein Akronym für *bootstrap aggregating*. Beim Bagging werden wiederholt Bootstrap-Stichproben (Stichproben mit Zurücklegen) aus dem Trainingsdatensatz gezogen (vgl. zu diesem Abschnitt Breiman 1996, S. 123). Auf Grundlage dieser Stichproben werden Zufallsvektoren und daraufhin die verschiedenen Entscheidungsbäume erstellt (vgl. Breiman 2001, S. 5).

Durch einige Vorteile sind Random Forests ansprechend für die Erstellung von Vorhersagen. Random Forests trainieren im Vergleich zu anderen Methoden sehr schnell und eignen sich sowohl für Regressions- als auch für Klassifikationsprobleme. Sie eignen sich ebenfalls für große Datensätze mit vielen Variablen und Beobachtungen. Außerdem basieren sie nur auf wenigen Hyperparametern (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 1).

3.2.1 Einführung in Entscheidungsbäume

Um Random Forests verstehen zu können, ist es wichtig, zuerst das Prinzip hinter Regressions- und Klassifikationsbäumen zu verstehen. Die in Random Forests verwendeten Bäume teilen den Prädiktorenraum mithilfe von binären Partitionen oder auch Teilungen (*Splits*) auf die jeweiligen Variablen. Der Prädiktorenraum ist ein p-dimensionaler Raum, der die p Attribute der Beobachtungen (x_1,x_2,...,x_p) umfasst (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 3).

Der Knoten am Anfang eines Baumes wird auch *Wurzel* genannt und umfasst den gesamten Prädikatorenraum. Die Knoten, die nicht geteilt werden, sind Endknoten und die finale Teilung des Prädikatorenraumes. Jeder nicht-finale Knoten wird in einen Unterknoten geteilt. Je nach Art des Schätzers geschieht dies auf eine andere Weise. Bei einem metrischen Schätzer wird ein Wert festgelegt und Punkte, die kleiner als dieser Wert sind, werden dem linken Unterknoten zugeordnet und der Rest dem rechten Unterknoten. Bei kategorischen Schätzvariablen existiert hier ein Unterschied. Diese können eine endliche Anzahl an Kategorien $S_i = \{s_{i,1},...,s_{i,m}\}$ annehmen. Eine Teilung weist dann eine Teilmenge der Kategorien $S \subset S_i$ dem linken und die restlichen Kategorien dem rechten Unterknoten zu (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 3). Schematisch lässt sich eine Teilung einer kategorischen Variablen wie in Abbildung 3.1 darstellen.

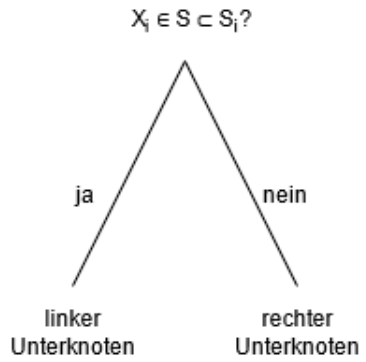

Abbildung 3.1: Teilung einer kategorischen Schätzvariable
Quelle: Eigene Darstellung, in Anlehnung an Cutler et al. (2011, S. 4)

Dieses Schema stellt eine einzige Teilung dar, die von anderen Teilungen bis zum finalen Knoten gefolgt wird.

Der Punkt, an dem die Teilung vorgenommen wird, wird gewählt, indem jede mögliche Aufteilung der verschiedenen Schätzvariablen in Betracht gezogen wird und danach die beste Aufteilung nach einem Kriterium ausgewählt wird. Für Regressionsbäume ist ein typisches Kriterium das mittlere quadratische Residuum am Knoten (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 3).

$$Q = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{3.10}$$

In (3.10) entspricht $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ dem vorhergesagten Wert am Knoten und ist der Durchschnitt der Zielwerte $y_1,...,y_n$. Bei Klassifikationsbäumen mit K Klassen hingegen ist ein Teilungskriterium der Gini Index, ein Maß für Ungleichheit.

$$Q = \sum_{k \neq k'}^{K} \hat{p}_k \hat{p}_{k'} \tag{3.11}$$

Dabei gibt $\hat{p}_k = \frac{1}{n}\sum_{i=1}^{n} I(y_i = k)$ den Anteil der Klasse k am Knoten an (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 3-4).

Diese Teilungskriterien geben bei Regressionsbäumen die Anpassungsgüte und bei Klassifikationsbäumen die Reinheit eines Knotens an. Je größer die Werte der Kriterien sind, desto schlechter die Anpassungsgüte und Reinheit. Im Folgenden gibt Q_L das Kriterium des linken Unterknotens und Q_R das Kriterium des rechten Knotens mit den Stichprobengrößen n_L und n_R, an. Die Teilung wird an dem Punkt vollzogen, an dem $Q_{split} = n_L Q_L + n_R Q_R$ minimiert wird. Bei kategorischen Variablen werden zudem, um die beste Teilung herbeizuführen, die Werte der Schätzer sortiert. Außerdem werden Q_L,Q_R und Q_{split} für alle Möglichkeiten, Teilmengen der Kategorien zuzuweisen, berechnet (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 4-5).

Nach der Teilung am Teilungspunkt wird die Prozedur auch mit den Unterknoten wiederholt, bis ein Stopkriterium erfüllt ist. Ein Kriterium dafür könnte sein, dass gestoppt wird, wenn nur noch eine bestimmte Anzahl von Fällen in den Endknoten auftreten. Nach dem Anhalten werden die geschätzten Werte aller Beobachtungen in den Endknoten ermittelt, in dem der Mittelwert der abhängigen Variablen bei einem Regressionsproblem und die häufigsten Klassen bei Klassifikationsproblemen berechnet werden (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 5).

3.2.2 Random-Forest-Algorithmus

Wie zu Beginn beschrieben, ist ein Random Forest ein Ensembleverfahren. Für einen p-dimensionalen Zufallsvektor $X = (X_1,...,X_p)^T$ wird eine gemeinsame Verteilung $P_{XY}(X,Y)$ angenommen. Der Zufallsvektor repräsentiert die tatsächlichen Eingangsvariablen und Y ist eine Zufallsvariable, die die echten Werte der abhängigen Variablen repräsentiert. Mit dem Random-Forest-Algorithmus soll eine Vorhersagefunktion $f(X)$, die Y vorhersagt, gefunden werden. Die Vorhersagefunktion kann dabei durch eine Verlustfunktion $L(Y,f(X))$ bestimmt werden, die den Erwartungswert des Verlustes minimiert (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 2).

Die Verlustfunktion gibt an, wie nah der vorhergesagte Wert $f(x)$ am tatsächlichen Wert Y ist und bestraft jene vorhergesagten Werte, die zu weit vom tatsächlichen Wert entfernt liegen. Für Regressionsprobleme lautet die Verlustfunktion oftmals

$$L(Y,f(X)) = (Y - f(X))^2 \tag{3.12}$$

und für Klassifikationsprobleme folgt dann (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 2):

$$L(Y,f(X)) = I(Y \neq f(X)) = \begin{cases} 0, & \text{wenn } Y = f(x) \\ 1, & \text{sonst} \end{cases} \tag{3.13}$$

Der Wert, der mit Random Forests geschätzt wird, wird auch Ensemble-Schätzer genannt. Die Idee der Ensemble-Methoden ist es, Basislerner $h_1(x)$,...,$h_J(x)$ zu kombinieren. Bei Klassifikationsproblemen, also wenn die zu schätzende Variable kategorisch ist, geschieht dies dadurch, dass die am häufigsten geschätzte Klasse als Ensemble-Schätzer gewählt wird. Für Klassifikationsprobleme lassen sich diese Mehrheitswahlen so formulieren:

$$f(x) = arg \max_{y \in \mathcal{Y}} \sum_{j=1}^{J} I(y = h_j(x)) \tag{3.14}$$

Hierbei steht $\mathcal{Y}$ für den Satz der möglichen Werte von Y. $I(\cdot)$ entspricht der Indikatorfunktion (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 2). $h_j(x)$ entspricht dem Vorhersagewert der abhängigen Variable unter Verwendung des j-ten Baumes an der Stelle x (vgl. Cutler et al. 2011, S. 8). Bei Regressionsproblemen entspricht der Ensemble-Schätzer dem Mittelwert der Basislerner.

Nun zum eigentlichen Algorithmus. Zu Beginn wird aus dem Trainingsdatensatz eine Bootstrap-Stichprobe der Größe N gezogen. Ein Baum wird daraufhin an eine Stichprobe angepasst. Alle Beobachtungen liegen dabei bei einem einzelnen Knoten. Anschließend wird unabhängig am Knoten die beste Teilung aus m zufällig ausgewählten Schätzvariablen ermittelt. Der Prozess der Ziehung von Stichproben, Anpassung der Bäume und Teilung wird fortgeführt, bis das Kriterium zum Anhalten erfüllt wurde. Die Vorhersagen der Bäume werden daraufhin aggregiert. Um eine Vorhersage an einem neuen Punkt x zu treffen, wird für die Klassifikation (3.14) und für Regression der Mittelwert der Basislerner verwendet (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 8).

3.2.3 Der Nutzen von Out-Of-Bag-Daten

Out-of-Bag-Daten sind jene Daten, die nicht durch das Bagging erfasst wurden (vgl. Cutler et al. 2011, S. 9). Etwa 36 % des Datensatzes, auf dem Bagging angewendet wurde, fällt in diese Kategorie (vgl. Bylander 2002, S. 289). Out-Of-Bag-Daten können als Validierungsdaten genutzt werden, da sie nicht im Trainingsdatensatz enthalten sind und zufällig gezogen wurden. Aus diesem Grund können Out-Of-Bag-Daten dazu genutzt werden, den Generalisierungsfehler, also die Fehlerrate beim Trainieren des Datensatzes, zu berechnen. Bei Regressionsproblemen geschieht dies für gewöhnlich über den mittleren quadratischen Fehler, wobei $\hat{f}_{oob}(x_i)$ die Out-Of-Bag-Schätzung für die i-te Beobachtung ist:

$$MSE_{oob} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{f}_{oob}(x_i)) \tag{3.15}$$

Für Klassifikationsprobleme kann die Fehlerrate durch

$$E_{oob} = \frac{1}{N} \sum_{i=1}^{N} I(y_i \neq \hat{f}_{oob}(x_i)) \tag{3.16}$$

beschrieben werden (vgl. zu diesem Abschnitt Cutler et al. 2011, S. 9). Mithilfe dieser Fehlerrate lässt sich die Vorhersagegütung der Schätzung ermitteln und kann damit auch zur Verbesserung des Random Forests beitragen. Beispielsweise lässt sich dadurch die optimale Anzahl der zufällig gezogenen Variablen m zu ermitteln.

3.2.4 Variablenwichtigkeit

Aufgrund dessen, dass jede Variable an verschiedenen Punkten und in verschiedenen Bäumen auftreten kann, wenn überhaupt, ist es nicht möglich, eine Art Durchschnittsbaum zu visualisieren. Dadurch sind Random Forests deutlich weniger leicht zu interpretieren als Entscheidungsbäume. Auf der anderen Seite haben Random Forests den Vorteil, dass jede Variable die Möglichkeit hat, in einen Baum aufgenommen zu werden. So lassen sich Informationen darüber gewinnen, wie stark der Einfluss der unabhängigen Variablen auf die zu schätzende Variable ist beziehungsweise welche der Variablen überhaupt relevant für die Schätzung sind. Dafür wird die Variablenwichtigkeit gemessen (vgl. zu diesem Abschnitt Strobl et al. 2009, S. 335).

Um die Variablenwichtigkeit zu ermitteln, können verschiedene Methoden angewendet werden. Eine Möglichkeit für Klassifikationsprobleme ist dabei beispielsweise die *Gini importance*. Diese basiert auf dem Gini index (3.11). Sie beschreibt die durchschnittliche Verbesserung der Ungleichheit des Gini-Kriteriums, die eine Variable an allen möglichen Punkten im Random Forest erreicht hat (vgl. zu diesem Abschnitt Strobl et al. 2009, S. 335).

Eine weitere oft verwendete Methode ist die *Permutationswichtigkeit*. Um die Variablenwichtigkeit für die k-te Variable zu bestimmen, werden für jeden Baum für die Out-of-Bag-Daten die Generalisierungsfehler (für Regressionsprobleme (3.15) und für Klassifikationsprobleme (3.16)) bestimmt. Die Werte der k-ten Variable werden zufällig permutiert und der Fehler wird anschließend für diese Werte erneut berechnet. Dann ist die Variablenwichtigkeit der Unterschied des Fehlers der permutierten Variable im Vergleich zum Fehler der nicht permutierten Variable. Die gesamte Variablenwichtigkeit wird errechnet in dem die Wichtigkeit der einzelnen Variablen über alle Beobachtungen gemittelt wird. Je geringer die Veränderung des Fehlers, desto geringer auch die Relevanz der Variable.

Wenn es keine Veränderung gibt, hat die Variable keinen Einfluss auf die unabhängige Variable (vgl. zu diesem Abschnitt Strobl et al. 2009, S. 335-336).

Formal lässt sich diese Variablenwichtigkeit wie folgt ausdrücken:

$$VI^{(t)}(X_k) = \frac{\sum\limits_{i \in \bar{B}^{(t)}} I(y_i \neq \hat{f}^{*(t)}(x_i)) - \sum\limits_{i \in \bar{B}^{(t)}} I(y_i \neq \hat{f}^{(t)}(x_i))}{|\bar{B}^{(t)}|} \tag{3.17}$$

In (3.17) seien $\bar{B}^{(t)}$ die Out-of-Bag-Daten der Bootstrap-Stichprobe für einen Baum t, mit $t \in \{1,...,n(t)\}$, und $|\bar{B}^{(t)}|$ sei die Kardinalität von $\bar{B}^{(t)}$. $\hat{f}^{*(t)}(x_i)$ sei die Schätzung für die i-te Beobachtung nach der Permutation der Werte der Variable k. Über die Mittelung über alle Bäume lässt sich dann die gesamte Variablenwichtigkeit für die Variable k bestimmen (vgl. zu diesem Abschnitt Strobl et al. 2009; Cutler et al. 2011, S. 335-336, S. 13):

$$VI(X_k) = \frac{1}{n(t)} \sum_{t=1}^{n(t)} VI^{(t)}(X_k) \tag{3.18}$$

Ein gutes Maß für die Wichtigkeit von Variablen ist der Unterschied in der Vorhersagegenauigkeit vor und nachdem die Variablen permutiert wurden, gemittelt über alle Bäume (Strobl et al. 2009, S. 335)

3.3 Künstliche neuronale Netze

In diesem Abschnitt werden künstliche neuronale Netze (KNN) vorgestellt. Die Idee hinter KNNs ist es, menschliches Denken nachzuahmen. Dabei wird das menschliche Nervensystem bestehend aus Nervenzellen (Neuronen) simuliert, um unter anderem Klassifikationsprobleme zu lösen und Muster zu erkennen. KNNs können von einem theoretischen Standpunkt aus jede mathematische Funktion lernen, sofern genügend Trainingsdaten vorhanden sind (vgl. Aggarwal 2018, S. VII). Neuronale Netze erfassen selbst subtile funktionale Beziehungen zwischen Daten, selbst wenn diese Beziehungen unbekannt und schwer zu beschreiben sind. Außerdem können KNNs verallgemeinern und Vorhersagen über zukünftiges Verhalten der Grundgesamtheit ableiten. Zudem sind sie universelle funktionale Schätzer und können flexiblere Formen als traditionelle Methoden annehmen. Damit eignen sie sich hervorragend für die Erstellung von Vorhersagemodellen (vgl. zu diesem Abschnitt Zhang et al. 1998, S. 35).

3.3.1 Aufbau

In dieser Arbeit wird sich auf vorwärtsgerichtete Netze konzentriert. Diese gehören zu den beliebtesten und am meistgenutzten Arten von KNNs.

Die Grundeinheiten von KNNs sind Neuronen. Jedes Neuron erhält eine Anzahl d Signale, die aus den Variablen x_1,...,x_d bestehen. Jedes Signal ist über ein Gewicht w_1,...,w_d mit dem Neuron verbunden. Die Signale werden mit den jeweiligen Gewichten multipliziert. Das Lernen in einem KNN geschieht über das Anpassen der Gewichte. Die Summe dieser gewichteten Signale lässt sich folgendermaßen ausdrücken (vgl. zu diesem Abschnitt Aggarwal 2018, S. 4-6):

$$\sum_{i=1}^{d} w_i x_i \tag{3.19}$$

Zu (3.19) wird ein Schwellenwert b oder auch *Bias* genannt hinzugefügt.

$$z = \sum_{i=1}^{d} w_i x_i + b \tag{3.20}$$

Die Ergebnisse eines Neurons werden stets nach Anwendung einer Aktivierungsfunktion angegeben (vgl. Aggarwal 2018, S. 12). Diese Funktion vergleicht die gewichtete Summe der Eingaben mit dem Schwellenwert b (vgl. Rojas 1996, S. 133). Für die Aktivierungsfunktion gibt es etliche Varianten. Für das Lösen eines binären Klassifikationsproblems eignet sich beispielsweise eine Schwellenwertfunktion. Wenn der Schwellenwert bei dieser überschritten wird, wird die Funktion aktiviert und ihr wird der Wert 1 (an) zugewiesen. Wird der Wert hingegen nicht überschritten, wird die Funktion nicht aktiviert und ihr wird der Wert 0 (aus) zugewiesen. Dabei entspricht $\hat{y}$ dem geschätzten Ausgabewert und $\varphi(z)$ sei die Aktivierungsfunktion der Gleichung z.

$$\hat{y} = \varphi(z) = \begin{cases} 1, & \text{wenn } z \geq 0 \\ 0, & \text{sonst} \end{cases} \tag{3.21}$$

Für Klassifikationsprobleme mit mehr als zwei Klassen eignet sich beispielsweise eine Sigmoid-Funktion:

$$\hat{y} = \varphi(z) = \frac{1}{1 + e^z} \tag{3.22}$$

Für die Erstellung der KNNs in dieser Arbeit wird die Tanh-Funktion verwendet:

$$\varphi(z) = \frac{e^{2z} - 1}{e^{2z} + 1} \tag{3.23}$$

Der Hauptunterschied zwischen der Sigmoid-Funktion und der Tanh-Funktion ist, dass Erstere Werte von $(0,1)$ und die Tanh-Funktion Werte von $(-1,1)$ annimmt. Das bedeutet zum Beispiel, dass die Aktivierungsfunktionen bei großen Werten von x Werte nahe 1

annehmen (vgl. zu diesem Abschnitt Aggarwal 2018, S. 12-13).

Mehrschichtige neuronale Netze bestehen, wie der Name schon vermuten lässt, typischerweise aus verschiedenen Schichten, die wiederum aus vernetzten Neuronen bestehen. Die erste Schicht ist die Eingabeschicht, die die externen Daten aufnimmt. Am Ende des KNNs ist die letzte Schicht, die Ausgabeschicht, welche die jeweilige Problemlösung ausgibt. Zwischen Eingabe- und Ausgabeschicht liegen eine oder mehrere Schichten in denen die Berechnungen stattfinden (vgl. zu diesem Abschnitt Zhang et al. 1998, S. 38). Diese Schichten werden versteckte Schichten genannt, da die Berechnungen innerhalb dieser Schichten für den Nutzer nicht einsehbar sind (vgl. Aggarwal 2018, S. 17). Ein Beispiel eines KNN mit zwei versteckten Schichten wird in Abbildung 3.2 dargestellt.

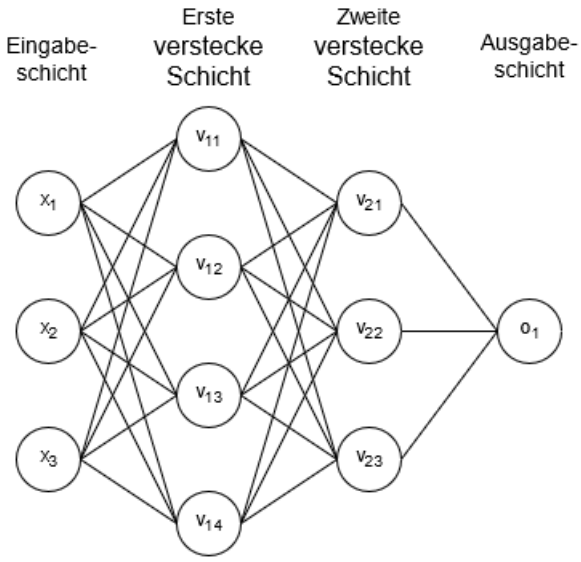

Abbildung 3.2: Aufbau eines zweischichtigen KNN
Quelle: Eigene Darstellung, in Anlehnung an Klinke (2021)

Das dargestellte KNN besteht aus einer Eingabeschicht mit drei Eingaben, einer versteckten Schicht mit vier Neuronen, einer zweiten versteckten Schicht mit drei Neuronen und einer Ausgabeschicht mit einem Ausgabeknoten. Die Anzahl der Knoten und Neuronen sind für jede Schicht frei wählbar.

Man unterscheidet die vorwärtsgerichteten Netze unter anderem an der Anzahl der versteckten Schichten. Flache neuronale Netze haben nur ein bis zwei versteckte Schichten, während tiefe KNNs drei oder mehr versteckte Schichten haben (vgl. Aggarwal 2018, S. 37). Letztere sind vor allem für sehr komplexe Probleme ausgelegt. Abbildung 3.2 zeigt demnach ein flaches KNN.

3.3.2 Lernprozess

Das Trainieren eines vowärtsgerichteten künstlichen neuronalen Netzes findet in dieser Arbeit über Rückpropagierung (engl. *backpropagation*) statt. Sie besitzt eine Vorwärts- und eine Rückwärtsphase (vgl. zu diesem Abschnitt Aggarwal 2018, S. 21). Die einzelnen Schritte des Algorithmus werden im folgenden Abschnitt dargestellt.

Vorwärtsphase Die Eingabewerte des Trainingssatzes werden dem KNN eingespeist, um die erste Schicht zu aktivieren. Jedes Neuron berechnet dann die Summe der gewichteten Aktivierungen der vorangegangen Schicht und fügt einen Bias zu. Dieser Prozess setzt sich bis zur Ausgangsschicht fort, welche dann das Ergebnis des KNNs ausgeben soll (vgl. zu diesem Abschnitt Aggarwal 2018, S. 21).

Ähnlich wie bei Random Forests wird mithilfe von Verlustfunktionen oftmals auch Fehlerfunktionen genannt, ermittelt, wie weit ein prognostiziertes Ergebnis vom tatsächlichen Wert entfernt ist. Für Regressionsprobleme ist eine häufig verwendete Funktion:

$$E_R = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{3.24}$$

Und für Klassifikationsprobleme:

$$E_C = \sum_{i=1}^{n} \sum_{j=1}^{C} y_{ij} log(\hat{y}_i) \tag{3.25}$$

Dabei sei $\hat{y}_i$ der vorhergesagte Wert und y_i der tatsächliche Wert an der Stelle i. C sei dabei die Anzahl der Klassen. Das Ziel dieser Funktionen ist es Gewichte zu finden, die E minimieren (vgl. zu diesem Abschnitt Klinke 2021).

Rückwärtsphase Um die optimalen Gewichte, die zu Beginn zufällig gewählt wurden, zu erhalten, wird das Konjugierte-Gradienten-Verfahren genutzt. Dabei wird die erste Ableitung der Fehlerfunktion nach den jeweiligen Gewichten berechnet, um die Gewichte zu ermitteln, die die Fehlerfunktion minimieren. Das Minimum wird mit den konjugierten Gradienten erreicht, indem die Gewichte iterativ aktualisiert werden. Diese Aktualisierungen lassen sich durch folgende Formel beschreiben (vgl. zu diesem Abschnitt Rojas 1996, S. 156-157):

$$w_{ij,t} = w_{ij,t-1} - \lambda_t \frac{\partial E}{\partial w_{ij}} \tag{3.26}$$

Dabei sei λ_t die Lernrate mit $\lim_{t \to \infty} = 0$ und w_{ij} sei das Gewicht des i-ten Eingabeknotens nach der j-ten versteckten Schicht oder Ausgabeschicht. Die Lernrate kontrolliert die

Geschwindigkeit der Fehlerminimierung. Ist die Rate zu gering besteht die Möglichkeit, dass die Phase an einem lokalen Minima endet und wenn sie zu groß ist, könnte das Ergebnis ungenau ausfallen (vgl. zu diesem Abschnitt Klinke 2021).

Nach Rojas (1996) stoppt der Algorithmus nach dem Minimieren der Fehlerupdates durch Aktualisierungen sobald der Wert der Fehlerfunktion ausreichend klein geworden ist.

3.3.3 Das Problem der Über- und Unteranpassung

Ein Modell mithilfe neuraler Netze an einen bestimmten Trainingsdatensatz anzupassen, garantiert nicht, dass die Vorhersage eines zuvor ungesehenen Testdatensatzes gut ist, selbst wenn die Vorhersage auf den Trainingssatz perfekt ist (vgl. Aggarwal 2018, S. 25). Ein Vohersagemodell sollte in der Lage sein zu verallgemeinern und das Ziel ist es dabei nicht, dass jeder einzelne Effekt erfasst wird. Dieses Problem wird Überanpassung genannt. Vorhersagemodelle mit diesem Problem sind möglicherweise nicht in der Lage zusätzliche Beobachtungen korrekt vorherzusagen.

Das Gegenteil ist beim Problem der Unteranpassung der Fall. Das KNN ist dann nicht in der Lage, Effekte mithilfe des Trainingsdatensatzes korrekt zu erfassen. Dadurch ist das Modell nicht in der Lage eine gute Vorhersage zu liefern.

Eine mögliche Lösung des Problems der Überanpassung ist es, Gewichtsverfall (engl. *weight decay*) einzuführen. Beim Rückpropagierungs-Algorithmus werden dabei bei jeder Iteration die Gewichte verringert. Dabei werden große Gewichte bestraft. Durch kleinere Gewichte wird die Aktivierungsfunktion linearer und dadurch simpler. Formal lässt sich der Gewichtsverfall als folgende Funktion ausdrücken:

$$E' = E + \omega(\sum w_j^2 + \sum w_{ij}^2) \tag{3.27}$$

In (3.27) sei ω der Verfallsparameter im Intervall $[0,1]$ (vgl. zu diesem Abschnitt Klinke 2021).

Ein weiterer Lösungsansatz ist das frühe Stoppen. Simpel ausgedrückt wird dabei die Fehlerrate des Netzes beim Validierungsdatensatz mit der Fehlerrate beim Trainingsdatensatz während des Lernprozesses miteinander verglichen. Ab einem Punkt wird die Fehlerrate beim Validierungssatz wieder steigen, selbst wenn sie weiterhin beim Trainingssatz sinkt. Wenn das Minimum der Fehlerrate beim Validierungsdatensatz erreicht ist, wird der Prozess gestoppt, da weitere Verbesserungen beim Trainingsdatensatz auf Überanpassung hindeuten (vgl. zu diesem Abschnitt Aggarwal 2018, S. 192).

3.4 Bewertungsmethoden

Um die Vorhersagemodelle, die mit verschiedenen Methoden erstellt wurden, im späteren Verlauf der Arbeit vergleichen zu können, werden verschiedene Bewertungsmethoden genutzt. Mithilfe dieser Methoden lassen sich unter anderem Aussagen darüber treffen, wie genau die Vorhersagen sind, die getroffen wurden.

3.4.1 Konfusionsmatrix und Genauigkeit

Konfusionsmatrizen eignen sich gut, um die Ergebnisse eines binären Klassifikationsmodells darzustellen. Diese Art der Matrix ist die Basis etlicher Bewertungsmetriken. Die Matrizen haben jeweils verschiedene mögliche Ergebnisse. Wenn der tatsächliche Wert positiv ist und das Modell dies korrekt vorhersagt, spricht man von *richtig positiv (RP)*. Wenn der tatsächliche Wert positiv ist, aber das Modell dies als negativ vorhersagt, nennt man dies *falsch positiv (FP)*. Wenn der tatsächliche Wert negativ ist und dieses Ergebnis vorhergesagt wird, nennt sich der Fall *richtig negativ (RN)*. Ist der vorhergesagte Wert dann jedoch positiv, weist man dem Fall ein *falsch negativ (FN)* zu. Die Matrix lässt sich demnach wie folgt darstellen (vgl. zu diesem Abschnitt Fawcett 2006, S. 862):

Tabelle 3.1: Beispiel einer Konfusionsmatrix

		Tatsächliche Klasse	
		positiv	negativ
Vorhergesagte Klasse	positiv	richtig positiv (RP)	falsch positiv (FP)
	negativ	falsch negativ (FN)	richtig negativ (RN)

Diese Matrix erlaubt es nun die Genauigkeit zu bestimmen. Diese gibt den Anteil der korrekt vorhergesagten Werte an der Gesamtanzahl der vorhergesagten Werte an und lässt sich formal mit

$$Genauigkeit = \frac{RP + RN}{RP + FP + FN + RN} \tag{3.28}$$

bestimmen (vgl. zu diesem Abschnitt Fawcett 2006, S. 862).

3.4.2 ROC-Kurve und AUC

Die *receiver operating characteristic curve (ROC-Kurve)* oder auch Grenzwertoptimierungkurve ist eine Methode, um Klassifizierungsmodelle basierend auf ihrer Leistung zu organisieren, visualisieren und auszuwählen (vgl. Fawcett 2006, S. 861).

ROC-Kurven sind zweidimensionale Graphen bei denen zum Beispiel die Richtig-Positiv-Raten (RP-Rate) auf der y-Achse und die Falsch-Positiv-Raten (FP-Rate) auf der

x-Achse abgetragen werden und stellen damit den relativen Kompromiss zwischen Vorteilen (richtig positiv) und Kosten (falsch positiv) dar. Die Raten lassen sich ebenfalls mithilfe der Konfusionsmatrix berechnen und sind in (3.29) und (3.30) dargestellt (vgl. zu diesem Abschnitt Fawcett 2006, S. 862).

$$RP\text{-}Rate = \frac{RP}{RP + FP} \tag{3.29}$$

$$FP\text{-}Rate = \frac{FP}{RN + FN} \tag{3.30}$$

Die RP-Rate gibt den Anteil der positiven Beobachtungen an, die korrekt klassifiziert wurden, während die FP-Rate den Anteil der negativen Beobachtungen angibt, die nicht korrekt klassifiziert wurden und falsch positiv sind. Die RP-Rate wird auch Sensibilität genannt. Weiterhin sei die Spezifität $1 -$ FP-Rate.

Eine Diagonale von (FP-Rate,RP-Rate)=(0,0) und (1,1) im Graphen gibt an, dass die RP-Raten gleich der FP-Raten sind und repräsentiert die Strategie, zufällig eine Klasse zu raten. Dabei würden bei einem ausgewogenen Datensatz demnach 50 % Beobachtungen richtig klassifiziert werden (vgl. zu diesem Abschnitt Fawcett 2006, S. 862-863).

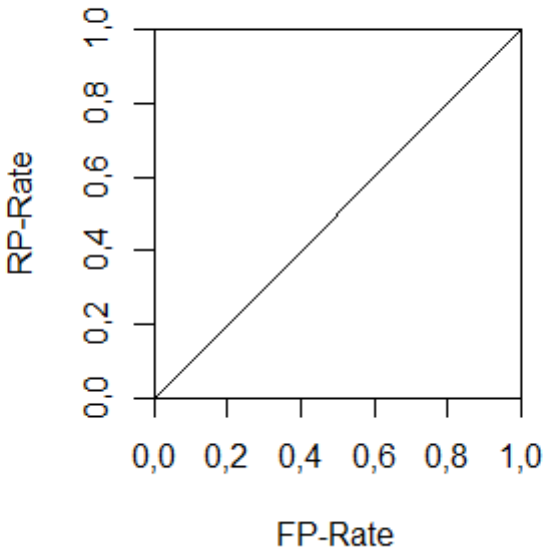

Abbildung 3.3: Diagonale im ROC-Raum
Quelle: Eigene Darstellung, in Anlehnung an Fawcett (2006, S. 862)

Die Abbildung 3.3 stellt diese Diagonale im quadratischen ROC-Raum dar. Der Punkt (0,0) gibt hier an, dass gar keine Beobachtungen als positiv klassifiziert werden. Damit gibt

es auch keine falschen positiven Schätzungen. Der Punkt (1,1) würde wiederum bedeuten, dass für alle Beobachtungen eine positive Klassifikation geschätzt worden wäre, auch wenn es negative Klassen gegeben hätte. Eine perfekte Klassifikation würde im Punkt (0,1) liegen, da dort alle Klassen korrekt geschätzt werden und es keine falschen positiven Klassen gibt. Punkte, die links dieser Diagonalen liegen, sind also besser als Punkte, die rechts dieser Diagonalen liegen (vgl. zu diesem Abschnitt Fawcett 2006, S. 862-863).

Um die verschiedenen ROC-Kurven schneller und leichter vergleichen zu können, wurde das Maß *area under curve (AUC)* eingefügt. Dieses steht für den Bereich unter der Kurve. Das Maß gibt den Anteil der Fläche unter der Kurve zur gesamten Fläche des ROC-Raums an und je größer der AUC-Wert ist, desto besser die Vorhersage. Eine Diagonale würde dabei den ROC-Raum halbieren und somit einen AUC-Wert von 50 % produzieren. Das bedeutet, dass ein realistisches Modell einen Wert von $\geq$ 50 % erreichen sollte. Eine wichtige Eigenschaft dieses Maßes ist, dass es gleich der Wahrscheinlichkeit ist, dass eine zufällig ausgewählte positive Instanz höher eingestuft wird als eine zufällig ausgewählte negative Instanz (vgl. zu diesem Abschnitt Fawcett 2006, S. 868).

4 Analyse

Das erste Vorhersagemodell, um die Stornierungen von Hotelbuchungen vorherzusagen, wird mit einer binären logistischen Regression erstellt.

4.1 Multiple binäre logistische Regression

4.1.1 Anpassung

Zu Beginn der Regressionsanalyse werden die unabhängigen Variablen auf Korrelationen untereinander überprüft. Der Grund dafür ist, dass eine abhängige Variable nicht als Funktion einer anderen Variable dargestellt werden sollte. Ist dies dennoch der Fall, tritt das Problem der Multikollinearität auf. Dass die abhängigen Variablen leicht untereinander korreliert sind, lässt sich meistens nicht vermeiden, jedoch können Variablen mit großen Korrelationen vom Modell ausgeschlossen werden, um dem Problem vorzubeugen. Sollte das Problem ignoriert werden, kann es das Regressionsmodell verzerren und falsche Ergebnisse liefern.

Um die Korrelationen der nominalskalierten abhängigen Variablen zu vergleichen, werden sie mit Cramer's V, einem Maß, das auf dem χ^2-Test basiert, berechnet und in Tabelle 4.1 in einer Korrelationsmatrix dargestellt.

Tabelle 4.1: Korrelationsmatrix der kategorischen abhängigen Variablen

	hotel	mahlz	mrktsg	vtrb	wdh_gast	rsrv_rm	anzahlen	kunde
hotel	1,00	0,24	0,15	0,19	0,05	0,32	0,18	0,05
mahlz	0,24	1,00	0,19	0,08	0,06	0,12	0,09	0,11
mrktsg	0,15	0,19	1,00	0,71	0,35	0,15	0,37	0,28
vtrb	0,19	0,08	0,71	1,00	0,30	0,11	0,09	0,08
wdh_gast	0,05	0,06	0,35	0,30	1,00	0,04	0,06	0,10
rsrv_rm	0,32	0,12	0,15	0,11	0,04	1,00	0,15	0,11
anzahlen	0,18	0,09	0,37	0,09	0,06	0,15	1,00	0,10
kunde	0,05	0,11	0,28	0,08	0,10	0,11	0,10	1,00

Die meisten der dargestellten Korrelationen sind tatsächlich unproblematisch und liegen unter 0,40. Ein Variablenpaar, bestehend aus den Variablen zum Marktsegment und Vertriebsweg, hat eine Korrelation von 0,71. Dieses Ergebnis ist nicht weiter verwunderlich, da einige der möglichen Variablenausprägungen vom Marktsegment und dem Vertriebsweg miteinander übereinstimmen. Im weiteren Verlauf der Analyse werden verschiedene

Regressionsmodelle erstellt, bei denen eine dieser Variablen jeweils ausgeschlossen wird. Danach werden sie miteinander verglichen, um zu identifizieren, welche Variable im finalen Modell auszuschließen ist.

Für die metrischskalierten unabhängigen Variablen werden die Korrelationen mithilfe der Bravais-Pearson-Korrelation berechnet und unter den Variablen gibt es keine problematischen Korrelationen. Der größte Teil der Korrelationen liegt unter 0,20 und der höchste Wert bei einem Variablenpaar (Anzahl der Kinder und Durchschnittskosten) liegt bei 0,34.

Für das Modell mit der Variablen zum Marktsegment und ohne der Variablen zum Vertriebsweg (Modell 1) und für das Modell mit der Variablen zum Vertriebsweg und ohne der Variablen zum Marktsegment (Modell 2) werden nun Rückwärtsregressionen durchgeführt. Bei Rückwärtsregressionen werden ausgehend von dem Modell mit allen Variablen solange Variablen ausgeschlossen, bis das Akaike-Informationskriterium minimiert wurde. Die Formel für die Berechnung dieses Kriteriums wird in (4.1) dargestellt, wobei k die Anzahl der Modellparameter sei (vgl. zu diesem Abschnitt Klinke 2021).

$$AIC = 2k - 2log(max\text{-}likelihood) \tag{4.1}$$

Dabei wird nach Ockhams Rasiermesser vorgegangen, denn die Komplexität des Modells wird mit der Güte des Modells abgewogen. Das Akaike-Informationskriterium ist ein relatives Gütemaß für das Modell und verschlechtert sich, je mehr Modellparameter dem Modell hinzugefügt werden.

Bei der Ausführung der Rückwärtsregression wird die Variable zur Anzahl der Tage auf der Warteliste entfernt. Außerdem wird zusätzlich die Variable zu den benötigten Parkplätzen entfernt, da diese in beiden Modellen nicht signifikant ist.

4.1.2 Modellauswahl

Nun, nachdem die beiden Modelle erstellt wurden, werden sie anhand der ROC-Kurven und der AUC-Werte miteinander verglichen, um das finale Modell auszuwählen. Dafür werden die beiden allgemeinen ROC-Kurven erstellt. Da bei einer allgemeinen ROC-Kurve kein genauer Schwellenwert festgelegt wird, wird die Kurve auf Basis aller Schwellenwerte erstellt. Das bedeutet, dass jeder Punkt auf der Kurve mit einem anderen Schwellenwert korrespondiert.

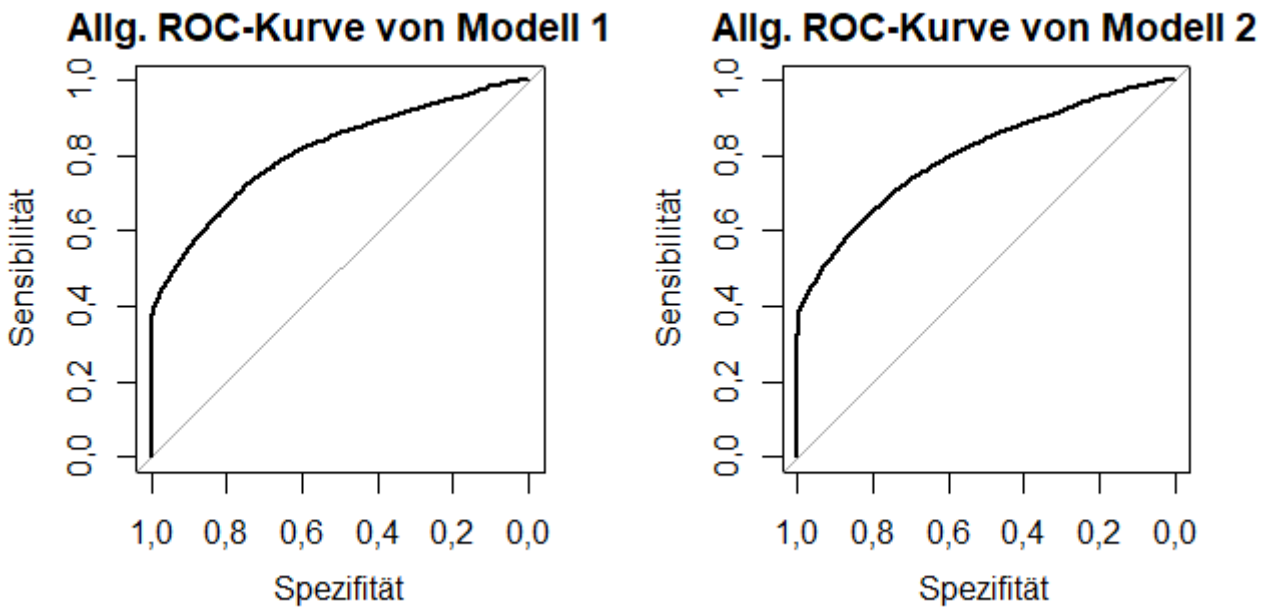

Abbildung 4.1: Allgemeine ROC-Kurven der beiden Logit-Modelle

Die beiden in Abbildung 4.1 dargestellten allgemeinen ROC-Kurven unterscheiden sich kaum. Jedoch spannt die ROC-Kurve des ersten Modells eine etwas größere Fläche auf. Modell 1 erreicht mit der allgemeinen ROC-Kurve einen AUC-Wert von 81,03 % und das zweite Modell einen AUC-Wert von 80,06 %.

Wie bereits am Anfang dieser Arbeit festgestellt wurde, ist der hier betrachtete Datensatz nicht ausgewogen. Bei ausgewogenen Datensätzen liegen die optimalen Schwellenwerte meistens nahe 0,5. Um die optimalen Schwellenwerte zu ermitteln, werden mit Hilfe der allgemeinen ROC-Kurven die Schwellenwerte so verändert, dass die AUC-Werte maximiert werden.

An dieser Stelle hätte man auch die Genauigkeit maximieren können, jedoch ist dabei das Problem, dass bei einem stark unausgewogenen Datensatz die Genauigkeit automatisch hoch ist, wenn alle Beobachtungen als die am häufigsten auftretende Klasse klassifiziert werden. Die Ergebnisse von ROC-Kurven sind bei unausgewogenen Datensätzen deshalb verlässlicher. Um das Problem der Unausgewogenheit zu umgehen, wäre eine weitere Option gewesen, einen Teildatensatz mit gleichgroßen Stichproben zu ziehen und auf Basis dieses Datensatzes die Genauigkeit zu maximieren.

Der ermittelte optimale Schwellenwert, der den AUC-Wert von Modell 1 maximiert, liegt bei etwa 0,3506 und für Modell 2 bei etwa 0,3617. Jedes Ergebnis der logistischen Regression, das über diesem Schwellenwert liegt, wird als *storniert* klassifiziert und jeder Wert, der gleich diesem Schwellenwert oder darunter liegt, wird als *nicht storniert* klassifizert.

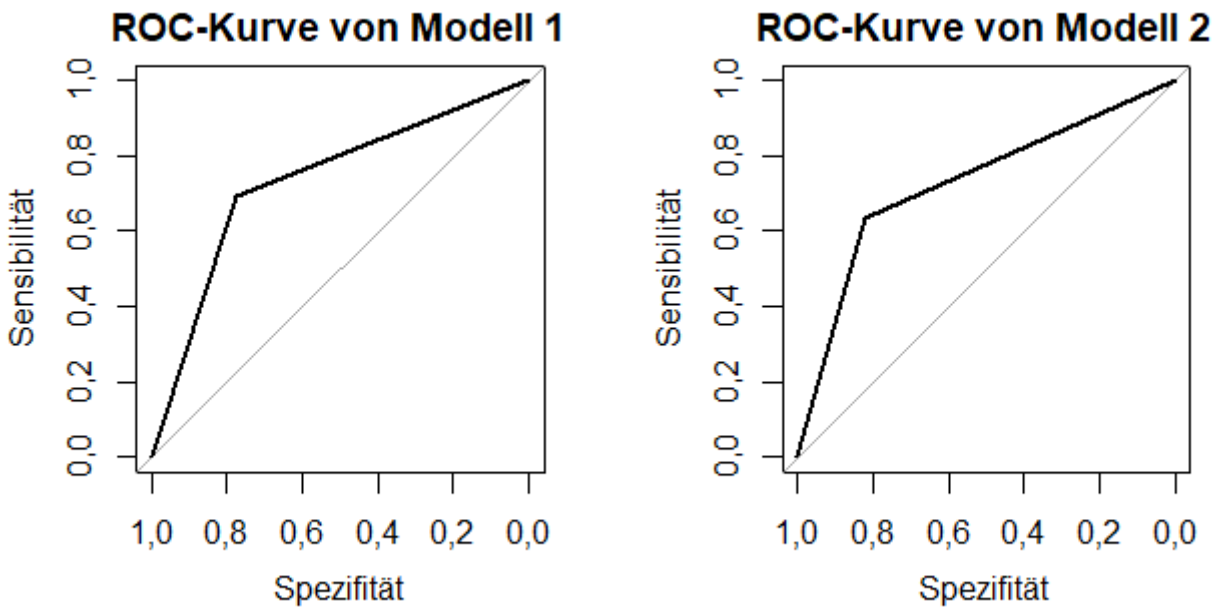

Abbildung 4.2: ROC-Kurven der beiden Logit-Modelle mit optimalen Schwellenwerten

Die ROC-Kurve des ersten Modells spannt erneut eine größere Fläche auf als die Kurve von Modell 2. Insgesamt erreicht das erste Modell einen AUC-Wert von 73,48 %, während das Modell 2 nur einen Wert von 72,71 % erreicht. Da der AUC-Wert von Modell 1 höher ist als der von Modell 2 wird das erste Modell als finales Regressionsmodell übernommen und in der weiteren Analyse verwendet.

4.1.3 Interpretation

Um die Ergebnisse des finalen logistischen Modells interpretieren zu können, werden sie in Tabelle 4.2 dargestellt.

Tabelle 4.2: Ergebnisse des finalen logistischen Regressionsmodells

	Koeffizient	Standardfehler	z-Wert	p-Wert
Konstante	-2,541	0,227	-11,22	<0,001
hotelResort	-0,215	0,0236	-9,134	<0,001
vorlauf	0,004	0,001	31,554	<0,001
erwachs	0,059	0,029	3,764	<0,001
kinder	0,095	0,029	3,332	<0,001
klnkndr	-0,574	0,116	-4,955	<0,001
mahlzVP	0,647	0,132	4,914	<0,001
mahlzHP	-0,168	0,033	-5,142	<0,001
mahlzKM	0,152	0,03	4,989	<0,001
mrktsgKomplement	-0,517	0,271	-1,91	0,056
mrktsgUnternehmen	-0,232	0,222	-1,042	0,297
mrktsgDirekt	-0,808	0,218	-3,712	<0,001
mrktsgGruppen	-0,06	0,219	-0,275	0,783
mrktsgReiseveranstalter	-0,672	0,217	-3,095	<0,001
mrktsgOnline	0,193	0,215	0,898	0,369
wdh_gast1	-0,689	0,104	-6,599	<0,001
vorher_storn	2,737	0,073	37,432	<0,001
vorher_n_storn	-0,619	0,037	-16,618	<0,001
rsrv_rmB	0,009	0,094	0,094	0,925
rsrv_rmC	0,128	0,16	1,203	0,229
rsrv_rmD	-0,004	0,027	-0,18	0,857
rsrv_rmE	0,039	0,042	0,923	0,356
rsrv_rmF	-0,155	0,068	-2,286	0,022
rsrv_rmG	0,019	0,077	0,258	0,797
rsrv_rmH	0,316	0,118	2,677	0,074
rsrv_rmL	1,615	1,496	1,079	0,28
rsrv_rmP	13,625	59,937	0,227	0,82
anzahlenNichtErstattbar	5,796	0,148	39,264	<0,001
anzahlenErstattbar	0,043	0,235	0,183	0,855
kundeGroup	0,059	0,203	0,291	0,771
kundeKurzfristig	0,931	0,066	14,094	<0,001
kundeKurzfristig_P	0,406	0,07	5,759	<0,001
dtr	0,003	0,001	11,483	<0,001
dauer	0,053	0,004	13,745	<0,001

Bei der Erstellung eines Regressionsmodells werden alle kategorischen Variablen als Dummyvariablen codiert. Die Klassen werden dabei zu Dummies. Wenn eine Klasse angenommen wird, nimmt der Dummy der Klasse den Wert 1 an und wenn die Klasse

nicht angenommen wird, den Wert 0. Der Konstanten β_0 werden dabei Referenzwerte zugeordnet, welche jeweils einer Klasse der kategorischen Variablen entsprechen. Die Referenzwerte sind jeweils die Klassen, die nicht in der Tabelle aufgeführt werden. Die Ergebnisse der Dummyvariablen werden dabei in Relation zu diesen Referenzwerten angegeben.

Jeder Koeffizient der metrischen Variablen, der in der Tabelle aufgeführt wird, ist hochsignifikant verschieden von 0. Die Signifikanz eines Koeffizienten kann am p-Wert gemessen werden. Koeffizienten sind signifikant, wenn der p-Wert kleiner als 0,1 sind. Je kleiner der p-Wert, desto signifikanter der Koeffizient. Dabei gelten Werte von <0,01 als hochsignifikant.

Auffällig ist, dass einige Koeffizienten der Dummyvariablen nicht signifikant sind und nicht bei der Modellerstellung ausgeschlossen wurden. Grund dafür ist, dass sich bei Ausschluss einer Klasse einer Dummyvariablen der Referenzwert der Konstanten und damit die Konstante selbst ändern kann. Koeffizienten von Klassen, die zuvor signifikant waren, könnten so insignifikant werden. Das bedeutet, dass selbst die insignifikanten Koeffizienten die signifikanten Koeffizienten der Klassen beeinflussen, weshalb sie nicht ausgeschlossen werden sollten.

Der Koeffizient der Konstanten nimmt den Wert -2,541 an, wenn jeweils die Referenzwerte angenommen werden und die Werte der metrischen Variablen gleich 0 sind. Mit der Formel e^{β}, wobei β der jeweilige Koeffizient sei, lassen sich alle Koeffizienten der Tabelle zu Chancenverhältnissen umwandeln (vgl. Sperandei 2014, S. 14).

Die Interpretation eines Koeffizienten einer Dummyvariablen wird hier anhand von *hotelResort*, also der Klasse *Resort* der Variablen *hotel* erläutert. Für *hotelResort* folgt, dass die Chance, dass eine Buchung bei einem Resort storniert wird, etwa 0,81-mal so hoch ist wie bei der Buchung eines Stadthotels, wenn die anderen Werte konstant bleiben. Grundsätzlich gilt bei kategorischen Variablen, dass positive Koeffizienten bedeuten, dass die Chance größer ist als beim Referenzwert und negative Koeffizienten bedeuten, dass die Chance geringer ist.

Anhand der Variablen *erwachs* wird im Folgenden die Interpretation der Konstanten von metrischen Variablen erläutert. Erneut wird der Koeffizient zu einem Chancenverhältnis umgewandelt. Jedoch wird für die Interpretation das Chancenverhältnis mit 1 subtrahiert $(e^{\beta} - 1)$ (vgl. Sperandei 2014, S. 15-16). Daraus folgt, dass wenn sich die Anzahl der Erwachsenen um 1 erhöht, steigt die Chance, dass die Buchung storniert wird um 6,08 %, wenn alle anderen Werte konstant bleiben. Wenn der Koeffizient positiv ist, steigt die Chance einer Stornierung und wenn sie negativ ist, sinkt Chance.

Ein interessanter Wert ist der Koeffizient zu den nicht erstattbaren Anzahlungen. Wenn die Anzahlung einer Buchung nicht erstattbar ist, ist die Chance, dass eine Buchung storniert wird, etwa 329-mal höher ist als bei Buchungen, die keine Anzahlung benötigen, wenn alle anderen Werte konstant bleiben. Intuitiv wäre zu erwarten gewesen, dass nicht

erstattbare Anzahlungen Anreize bieten, eine Buchung nicht zu stornieren und die Chance einer Stornierung geringer wäre als bei einer Buchung ohne Anzahlung. Dies scheint bei der logistischen Regression jedoch nicht der Fall zu sein.

4.2 Random Forest

Nach der Erstellung und Interpretation des finalen Regressionsmodells, wird sich nun den Random Forests zugewendet.

4.2.1 Wahl der Hyperparameter und des Schwellenwertes

Die Wahl der Hyperparameter beeinflusst die Leistung von Random Forests stark. Hyperparameter sind Parameter, die das Trainieren eines Modells kontrollieren. In diesem Abschnitt wird sich auf die Anzahl der Variablen, die von jedem Baum zufällig gezogen werden und die Anzahl der Bäume konzentriert.

Die Ermittlung der optimalen Variablenanzahl erfolgt durch Zuhilfenahme eines Algorithmus und der Out-Of-Bag-Fehler. Bei diesem Algorithmus werden ausgehend vom Modell mit zwei Variablen die Out-Of-Bag-Fehler überprüft. Zuerst wurden Modelle mit weniger als zwei Variablen überprüft und danach Modelle mit mehr als zwei Variablen. Der Algorithmus stoppt, sobald der Out-Of-Bag-Fehler nicht mehr kleiner wird und wieder ansteigt. Dieser Algorithmus wird auf Random Forests mit 500, 750 und 1000 Bäumen angewendet. Je nach Anzahl der Bäume kann die optimale Variablenanzahl variieren.

Für die Random Forests lassen sich damit Graphen erstellen, an dem die optimale Variablenanzahl abzulesen ist. In Abbildung ist der Graph für das Modell mit 500 Bäumen als Beispiel dargestellt.

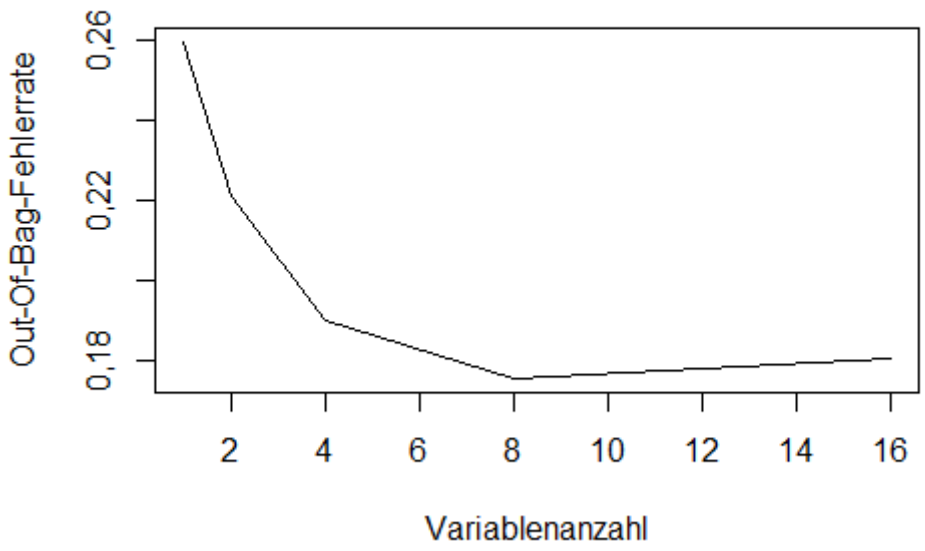

Abbildung 4.3: Out-Of-Bag-Fehlerrate nach Variablenanzahl des Modells mit 500 Bäumen

Bei einer Variablenanzahl von acht wird der Out-Of-Bag-Fehler minimiert. Die Fehlerrate liegt dabei bei 17,45 %. Durch den Algorithmus wurden Anzahlen der Variablen von 1, 2, 4, 8 und 16 überprüft. Die Fehlerraten bei den Variablenanzahlen sieben und neun sind jeweils größer als bei einer Anzahl von acht.

Nach (Cutler et al. 2011) hat eine Erhöhung der Baumanzahl keine negativen Auswirkungen auf die Vorhersageleistung eines Random Forests. Jedoch nimmt der Leistungszugewinn mit steigender Baumanzahl ab. Ein Nachteil von einer hohen Baumanzahl ist, dass die benötigte Rechenleistung ebenfalls hoch ist und somit auch die Berechnungsdauer des Random Forests mit jedem Baum steigt.

Tabelle 4.3: Ergebnisse der Random Forests

	Variablen	Bäume	allg. AUC-Wert	AUC-Wert mit Schwelle
Model 1	8	500	89,78 %	80,78 %
Modell 2	8	750	89,79 %	80,74 %
Modell 3	8	1000	89,79 %	80,68 %

Die optimale Variablenanzahl der verschiedenen Random Forests liegt jeweils bei acht. Der allgemeine AUC-Wert des dritten Modells ist etwas größer als der Wert von Modell 2 und Modell 1. Interessanterweise ist es beim AUC-Wert mit Schwellenwert umgekehrt. Dort erreicht das Modell mit 500 Bäumen einen höheren Wert als die anderen beiden

Modelle. Der optimale Schwellenwert des ersten Modells liegt bei 0,375. Das erste Modell wird aufgrund der schnellen Berechnung und der sich kaum von den anderen Modellen unterscheidenden ROC-Werte als finales Modell gewählt. Wie zuvor gibt die allgemeine ROC-Kurve die Kurve für jeden möglichen Schwellenwert an und spannt deswegen eine größere Fläche auf als die ROC-Kurve für einen bestimmten Schwellenwert.

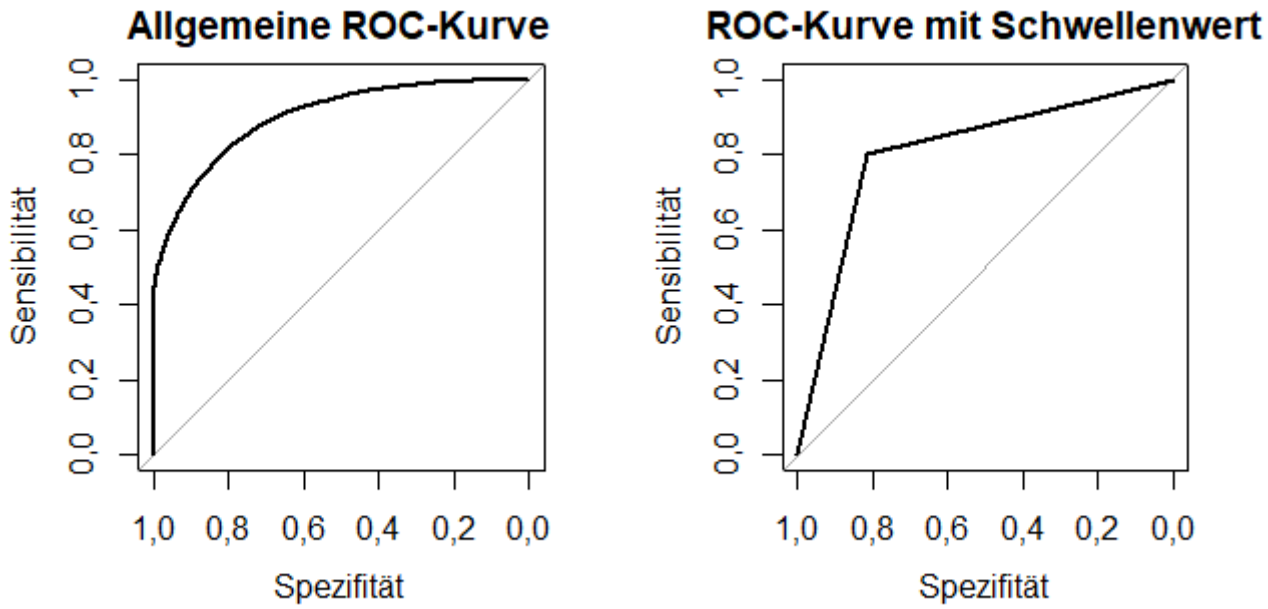

Abbildung 4.4: ROC-Kurven von Modell 1

4.2.2 Variablenwichtigkeit

Die wichtigsten Variablen werden in diesem Abschnitt mit der Permutationswichtigkeit identifiziert. Dafür wurde der Wichtigkeitswert in folgender Abbildung für jede Variable abgebildet.

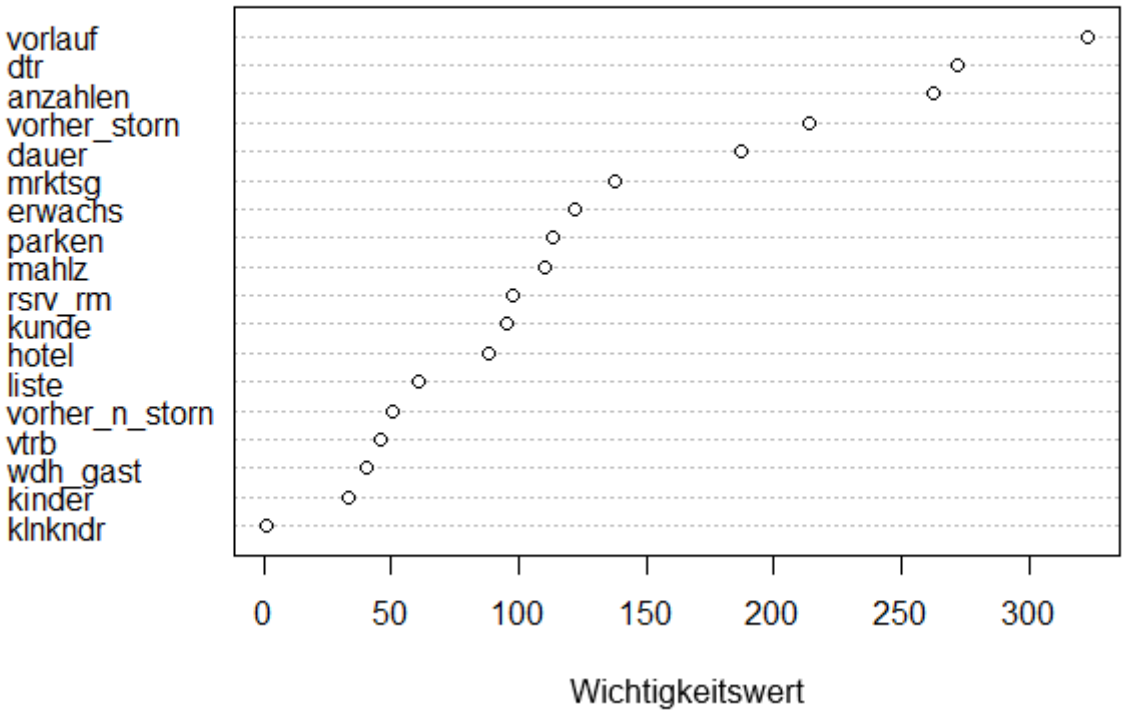

Abbildung 4.5: Variablenwichtigkeiten des finalen Random Forests

In der Abbildung 4.5 werden auf der y-Achse die Variablen absteigend nach Wichtigkeit abgebildet. Je höher der Wichtigkeitswert auf der x-Achse ist, desto wichtiger ist die Variable. Dabei sind die Variablen *vorlauf*, *dtr*, *anzahlen*, *vorher_storn* und *dauer* die fünf Wichtigsten. Das bedeutet, dass ein Ausschluss einer dieser Variablen den größten Genauigkeitsverlust bewirkt. Dass die Variablen *anzahlen* und *vorher_storn* eine hohe Wichtigkeit haben, lässt sich zum Beispiel dadurch erklären, dass Personen durch nicht erstattbare Anzahlungen Anreize haben, eine keine Stornierung zu vollziehen. Außerdem könnte eine hohe Anzahl vorheriger Stornierungen mit einer erhöhten Wahrscheinlichkeit einer Stornierung zusammenhängen.

Die drei Variablen, die am unwichtigsten sind, sind *wdh_gast*, *kinder* und *klnkndr*. Ein Ausschluss dieser Variablen scheint demnach keinen großen Informationsverlust beim Random Forest Modell zu bewirken.

4.3 Künstliche neuronale Netze

Bei der Erstellung der künstlichen neuronalen Netze werden die metrischen Variablen normalisiert beziehungsweise standardisiert. Die Variablen zu normalisieren, kann zu

besseren Ergebnissen der KNNs führen, da sich die Verteilungen verschiedener Variablen stark unterscheiden können. Das Problem dabei ist, dass die Verlustfunktion in solchen Fällen eine Tendenz hat sensibel auf einige Variablen zu sein. Ziel ist also, dass die Netze weniger sensibel auf die unterschiedlichen Verteilungen der Variablen sind. Eine Möglichkeit der Normalisierung ist die Min-Max-Normalisierung. Die Werte werden dabei so skaliert, dass sie im Bereich (0,1) liegen (vgl. zu diesem Abschnitt Aggarwal 2018, S. 127).

$$z = \frac{x_{ij} - min_j}{max_j - min_j} \tag{4.2}$$

Dabei seien min_j und max_j jeweils die minimalen und maximalen Werte des j-ten Attributs. Weiterhin sei x_{ij} der Variablenwert der j-ten Dimension an der Stelle i.

Die für die Erstellung dieser KNNs genutzte Normalisierung ist die sogenannte Standardisierung.

$$Z = \frac{X - \mu}{\sigma} \tag{4.3}$$

X sei die Variable, μ der Erwartungswert von X und σ die Standardabweichung von X. Das Konzept hinter der Standardisierung ist, dass angenommen wird, dass jedes Merkmal einer Standardnormalverteilung entnommen wird. Diese hat eine Null als Mittelwert und eine Eins als Standardabweichung (vgl. zu diesem Abschnitt Aggarwal 2018, S. 127).

4.3.1 Hyperparameteroptimierung

Eine zentrale Frage bei KNNs ist es, welche Hyperparameterwerte die Modellvorhersagen optimieren. Um diese Frage zu beantworten, werden in vielen Fällen Rastersuchen vorgenommen. Bei Rastersuchen werden zuvor festgelegte Hyperparameterwerte miteinander kombiniert und daraus KNNs erstellt. Das Problem der Rastersuche wird spätestens dann offensichtlich, wenn viele Hyperparameter mit vielen Werten festgelegt wurden. Mit jedem weiteren Wert im Raster nimmt die Anzahl der Kombinationen und der zu erstellenden KNNs exponentiell zu (vgl. zu diesem Abschnitt Aggarwal 2018, S. 125-126). Die benötigte Rechenleistung rechtfertigt den Leistungszugewinn dabei oftmals nicht.

Eine Lösung für dieses Problem kann die Zufallssuche sein. Dabei werden KNNs auf Basis von zufällig aus einem Raster ausgewählten Hyperparameterkombinationen erstellt. Bergstra et al. (2012) stellten dabei fest, dass die Zufallssuche genauso gute, wenn nicht sogar bessere Ergebnisse liefert als eine Rastersuche und das in einem Bruchteil der Zeit, die eine vollständige Rastersuche benötigt hätte.

Das finale einschichtige KNN wird nun mithilfe einer Zufallssuche bestimmt und die genutzte Aktivierungsfunktion ist die Tanh-Funktion. Die verwendeten Hyperparameter

sind die Anzahl der Neuronen, die Anzahl der Epochen, der Gewichtsverfall und die Lernrate. Die Anzahl der Epochen bestimmt, wie oft die Vorwärts- und Rückwärtsphase durchlaufen wird und bestimmt damit, wie oft die Gewichte aktualisiert werden.

Tabelle 4.4: Werte der Hyperparameter

	Hyperparameterwerte
Neuronen	5; 10; 20; 30
Epochen	1; 3; 5; 7; 10
Gewichtsverfall	0,0001; 0,001; 0,01; 0,1
Lernrate	0,0001; 0,001, 0,01; 0,1

Aus diesen Werten lassen sich nun 320 verschiedene Kombinationen erstellen. Die Zufallssuche stoppt, wenn sich der AUC-Wert, basierend auf der Vorhersage des Validierungsdatensatzes der erstellten Modelle nach fünf Modellen nicht um 0,01 Prozentpunkte verbessert oder wenn 40 Modelle erstellt wurden. Im Anschluss werden die erstellten Modelle nach dem AUC-Wert sortiert und es wird das Modell mit dem höchsten Wert ausgewählt.

Tabelle 4.5: Hyperparameterwerte der fünf Modelle mit den höchsten AUC-Werten

	Neuronen	Epochen	Gewichtsverfall	Lernrate	allg. AUC-Wert
Modell 1	20	7	0,0001	0,1	83,63 %
Modell 2	30	1	0,001	0,001	83,2 %
Modell 3	10	1	0,001	0,001	82,75 %
Modell 4	10	1	0,001	0,1	82,36 %
Modell 5	20	7	0,001	0,001	82,31 %

Bei der Zufallssuche werden 40 Modelle aus verschiedenen Kombinationen der Hyperparameterwerte erstellt. Das bedeutet, dass sich die AUC-Werte erstellten Modelle jeweils nach fünf Modellen um mindestens 0,01 Prozentpunkte verbessert haben. Demnach werden 12,5 % der 320 möglichen Kombinationen genutzt. Das Modell 1 ist das Modell mit dem höchsten AUC-Wert und der Wert ist um etwa 0,43 Prozentpunkte größer als der Wert des zweitbesten Modells. Es basiert auf 20 Neuronen in der versteckten Schicht und die Gewichte werden insgesamt sieben Mal aktualisiert. Die Werte des Gewichtsverfalls und der Lernrate betragen 0,0001 und 0,1. Der allgemeine AUC-Wert für den Validierungsdatensatz beträgt 83,63 %. Das Modell mit Schwellenwert erreicht einen AUC-Wert von 75,85 %.

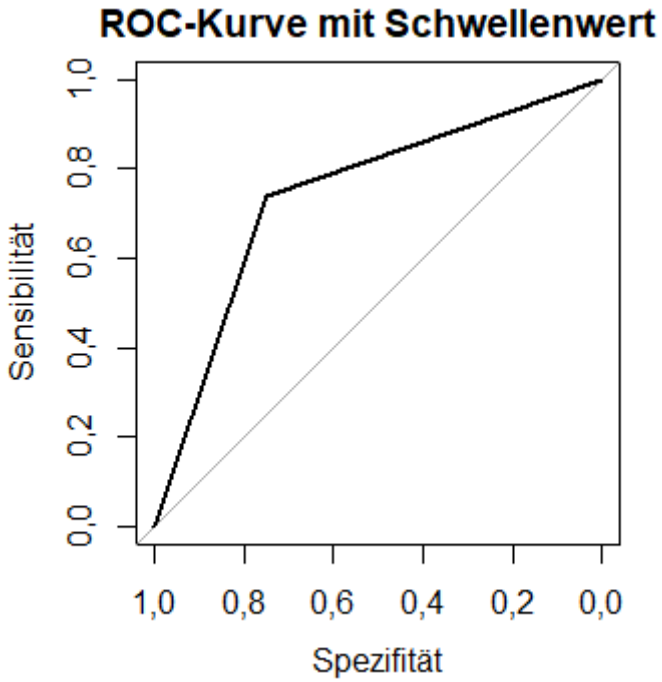

Abbildung 4.6: ROC-Kurve des finalen neuronalen Netzes

5 Vergleich der verschiedenen finalen Modelle

Nachdem jeweils die optimalen Modelle für die logistische Regressionen, die Random Forests und die KNNs ermittelt wurden, wird nun die Leistung der Modelle miteinander verglichen. Um das zu tun, wird der bisher ungesehene Testdatensatz herangezogen und für ihn werden durch die drei finalen Modelle Vorhersagen getroffen. Auf Grundlage dieser Vorhersagen werden anschließend ROC-Kurven und AUC-Werte berechnet. Zusätzlich wird außerdem die Genauigkeit betrachtet, die jedoch eine untergeordnete Rolle einnimmt, da der Datensatz, wie zuvor angemerkt wurde, unausgewogen ist.

Tabelle 5.1: Vergleich der Vorhersageleistungen der Modelle

	Allg. AUC-Wert	AUC-Wert mit Schwellenwert	Genauigkeit
Logistische Regression	81,14 %	73,42 %	74,1 %
Random Forest	89,59 %	80,35 %	81,71 %
KNN	83,5 %	75,48%	75,45 %

Für die Vorhersage der Hotelbuchungsstornierungen erreicht das Modell des Random Forests die beste Leistung. Sowohl der allgemeine AUC-Wert als auch der AUC-Wert mit Schwellenwert sind mit 89,59 % und 80,35 % deutlich höher als die Werte der anderen. Es

erreicht außerdem die beste Genauigkeit und es wird nur der Stornierungsstatus von 18,29 % der Buchungen inkorrekt vorhergesagt. Die Methode mit der zweitbesten Leistung ist das KNN.

6 Zusammenfassung und Fazit

Das Ziel dieser Arbeit war es, verschiedene Vorhersagemodelle zu Hotelbuchungsstornierungen zu erstellen und miteinander zu vergleichen. Der Datensatz wurde dafür zu Beginn auf unmögliche Werte untersucht. Da keine Werte der Beobachtungen unmöglich oder stark unrealistisch waren, wurden keine Beobachtungen von der Analyse ausgeschlossen. Die Werte des betrachteten Datensatzes entsprechen reellen Daten. Es ist zwar unwahrscheinlich, dass Hotelzimmer für beispielsweise 70 Tage gebucht werden, jedoch ist der Vorteil maschinellen Lernens, dass die Modelle auf Basis dieser Werte dennoch gute Vorhersagen treffen können.

Für die logistische Regression wurden die Variablen auf problematische Korrelationen überprüft. Dabei viel auf, dass die Variablen *mrktsg* und *vtrb* stark miteinander korrelieren. Die restlichen Variablen waren unbedenklich hinsichtlich der Multikollinearität. Im Anschluss daran wurden Rückwärtsregressionen durchgeführt, bei denen die nicht signifikanten Parameter vom Modell ausgeschlossen wurden. Bei der Erstellung des optimalen Random Forests wurde sich auf die optimale Variablenanzahl und die Anzahl der Bäume konzentriert. Dabei wurde ermittelt, dass die optimale Variablenanzahl für 500 bis 1500 Bäume pro Random Forest bei acht liegt. Da sich die Genauigkeit kaum verbesserte und der ROC-Wert sich sogar verschlechterte, wurde sich für das Modell mit 500 Bäumen entschieden. Bei der Erstellung der KNNs wurde sich aus Zeitgründen auf eine versteckte Schicht beschränkt. Es wurde eine zufällige Rastersuche angewendet, weil diese die Anzahl der Modelle aus den Kombinationen verringert. Das optimale Modell bestand in diesem Fall aus 20 Neuronen und durchlief insgesamt sieben Epochen.

Die Frage nach dem besten Modell muss aus verschiedenen Blickwinkeln betrachtet werden. Wenn es um die reine Vorhersageleistung geht, ist das Modell des Random Forests im Fall dieser Arbeit im Vorteil. Nicht nur, dass die AUC-Werte größer sind als die der anderen Modelle, sondern es erreicht außerdem eine Genauigkeit von 81,71 %. Damit ist die Genauigkeit des Random Forests um mehr als 6 Prozentpunkte besser als das des KNNs. Das bedeutet, dass das finale Modell des Random Forests zwar keine perfekten Ergebnisse liefert, aber es bietet die sichersten Vorhersagen aus den drei erstellten Modellen.

Auf der anderen Seite ist das Modell des Random Forests schlechter für Interpretationen geeignet. Wenn die Interpretierbarkeit das wichtige Maß für die finale Entscheidung ist, dann hat die logistische Regression klare Vorteile. Die Erstellung eines Logit-Modells ist simpler und intuitiver als die Erstellung der anderen Modelle und der Einfluss jeder

Variablen lässt sich interpretieren. Es lassen sich Aussagen treffen darüber, ob eine Variable überhaupt einen signifikanten Einfluss auf die Ergebnisse des Modells hat. Beim Logit-Modell in dieser Arbeit lässt sich allein am Vorzeichen der Koeffizienten ablesen, ob eine Variable einen positiven oder einen negativen Einfluss auf Hotelbuchungsstornierungen haben. Außerdem lässt sich die Größe des Einflusses dabei berechnen. Ein weiterer Vorteil der logistischen Regressionen ist, dass die Berechnungsdauer deutlich kürzer ist als die der anderen Modelle.

An dieser Stelle muss angemerkt werden, dass die Ergebnisse zur Vorhersageleistung keinesfalls für jede Vorhersage gelten. Random Forests liefern nicht immer bessere Ergebnisse als logistische Regressionen und KNNs. Letztere haben den Vorteil, dass sie so gut wie jede mathematische Formel lernen können und so breitere Anwendungsgebiete finden. Außerdem wurden in dieser Arbeit lediglich simple, flache KNNs genutzt. Komplexere, tiefe KNNs hätten womöglich noch bessere Ergebnisse erzielen können als die Random Forests, wobei die benötigte Rechenleistung stark ansteigen kann.

Zum Schluss lässt sich demnach feststellen, dass es keine pauschale Antwort auf die Frage zum besten Vorhersagemodell gibt. Jedes der erstellten Modelle besticht in einem anderen Aspekt und hat auch Nachteile. Jedes dieser Modelle ist gut dazu geeignet, die Hotelbuchungsstornierungen vorherzusagen. Damit treten die Stornierungen nicht zufällig, sondern strukturell auf.

Dass trotz der Simplizität der in dieser Arbeit erstellten Modelle gute Ergebnisse erzielt werden konnten, könnte bedeuten, dass eine Implementation von maschinellem Lernen in der Entscheidungsfindung bei Hotels lohnenswert wäre. Komplexe Modelle könnten bessere Ergebnisse erzielen und tatsächlich erstellten Antonio, De Almeida et al. (2017) mit einem ähnlichen Datensatz Modelle, die Genauigkeiten von teilweise 98,6 % aufwiesen. Hotels könnten bei bedenklichen Buchungen Überbuchungen einkalkulieren und somit Gewinne weiter maximieren. Damit könnte Anzahl der Überbuchungen reduziert und diese zielgerichteter eingesetzt werden. Die Vorhersageleistung der verwendeten Modelle hätte damit einen direkten Einfluss auf die Einnahmen der Hotels. Dadurch könnte maschinelles Lernen die Lösung der durch die Stornierungen verursachten Probleme sein, da bei perfekten Vorhersagen Kunden Überbuchungen nicht mehr wahrnehmen, weil sie ihr gebuchtes Zimmer erhalten.

Abkürzungsverzeichnis

KNN künstliche neuronale Netze

ROC-Kurve receiver operating characteristic curve

AUC area under curve

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur

Aggarwal, Charu C. (2018). *Neural Networks and Deep Learning*. Springer. DOI: 10.1007/978-3-319-94463-0.

Antonio, Nuno, Ana De Almeida und Luís Nunes (Apr. 2017). "Predicting Hotel Booking Cancellation to Decrease Uncertainty and Increase Revenue". In: *Tourism and Management Studies* 13, S. 25–39. DOI: 10.18089/tms.2017.13203.

Antonio, Nuno, Ana de Almeida und Luis Nunes (2019). "Hotel booking demand datasets". In: *Data in Brief* 22, S. 41–49. DOI: 10.1016/j.dib.2018.11.126.

Bergstra, James und Y. Bengio (2012). "Random Search for Hyper-Parameter Optimization". In: *The Journal of Machine Learning Research* 13, S. 281–305. DOI: 10.5555/2188385.2188395.

Breiman, Leo (1996). "Bagging Predictors". In: *Machine Learning* 24, S. 123–140. DOI: 10.1023/A:1018054314350.

– (2001). *Random Forests*, S. 5–32. DOI: 10.1023/A:1010933404324.

Bylander, Tom (2002). "Estimating Generalization Error on Two-Class Datasets Using Out-of-Bag Estimates". In: *Machine Learning* 48, S. 287–297. DOI: 10.1023/A:1013964023376.

Cutler, Adele, David Cutler und John Stevens (2011). *Random Forests*. Bd. 45, S. 157–176. DOI: 10.1007/978-1-4419-9326-7_5.

Fawcett, Tom (2006). "An introduction to ROC analysis". In: *Pattern Recognition Letters*, S. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

Hosmer Jr, David W, Stanley Lemeshow und Rodney X Sturdivant (2013). *Applied logistic regression*. Bd. 398. John Wiley & Sons. DOI: 10.1002/9781118548387.

Klinke, Sigbert (2021). *Vorlesungsfolien Datenanalyse II*.

Mostipak, Jesse (2020). *Kaggle.com Hotel booking demand*. https://www.kaggle.com/jessemostipak/hotel-booking-demand. Letzter Zugriff am 12.06.2021.

Rebala, Gopinath, Ajay Ravi und Sanjay Churiwala (2019). *An Introduction to Machine Learning*. Springer. DOI: 10.1007/978-3-030-15729-6.

Rojas, Raúl (1996). *Neural Networks: A Systematic Introduction*. Springer-Verlag. DOI: 10.1007/978-3-642-61068-4.

Romero Morales, Dolores und Jingbo Wang (2010). "Forecasting Cancellation Rates for Services Booking Revenue Management Using Data Mining". In: *European Journal of Operational Research* 202, S. 554–562. DOI: 10.1016/j.ejor.2009.06.006.

Sperandei, Sandro (2014). "Understanding logistic regression analysis". In: *Biochemia medica* 24, S. 12–8. DOI: 10.11613/BM.2014.003.

Strobl, Carolin, James Malley und Gerhard Tutz (2009). “An Introduction to Recursive Partitioning: Rationale, Application, and Characteristics of Classification and Regression Trees, Bagging, and Random Forests”. In: *Psychological methods* 14, S. 323–348. DOI: 10.1037/a0016973.

Toh, Rex S., Mary J. Rivers und Teresa W. Ling (2005). “Room occupancies: cruise lines out-do the hotels”. In: *International Journal of Hospitality Management* 24, S. 121–135. DOI: https://doi.org/10.1016/j.ijhm.2004.05.005.

Zhang, Peter, Eddy Patuwo und Michael Hu (1998). “Forecasting With Artificial Neural Networks: The State of the Art”. In: *International Journal of Forecasting* 14, S. 35–62. DOI: 10.1016/S0169-2070(97)00044-7.

Eigenständigkeitserklärung

Hiermit erkläre ich, Julius Freidank, dass ich die vorliegende Arbeit noch nicht für andere Prüfungen eingereicht habe. Ich habe die Arbeit selbständig verfasst. Sämtliche Quellen einschließlich Internetquellen, die ich unverändert oder abgewandelt wiedergegeben habe, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, habe ich als solche kenntlich gemacht. Ich bin mir darüber bewusst, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.