Dokumentation Gruppe 3 kNN-Algorithmus

Projekt KI

An der Dualen Hochschule Baden-Württemberg Heidenheim

in der Fakultät Wirtschaft

im Studiengang Wirtschaftsinformatik

eingereicht von

Jannik Maier

Leonie Fetzer

Lena Bäurle

Link zum Code GitHub:

https://github.com/LeonieFetzer/DHBW_kNN_Gruppe-3

Semester 2023/5

Abgabedatum: 15.09.2025

Dozent Dr. Höchenberger

Gliederung

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

kNN	k-Nearest-Neighbors
SVM	Support Vector Machine

1. Einleitung

Im Rahmen des Projektes im 5. Semesters im Studiengang Wirtschaftsinformatik der DHBW Heidenheim, wird sich mit unterschiedlichen Algorithmen befasst.

Die Aufgabenstellung umfasst die Bearbeitung eines Datensatzes, welcher Informationen zu Kunden enthält, welchen per Telefon Produkte verkauft wurden. Die Bearbeitung des Telefonkunden-Datensatzes soll mit dem k-Nearest-Neighbors(kNN)-Algorithmus bearbeitet werden. Hierfür wird zunächst der Datensatz betrachtet, ehe anschließend Fragestellungen dazu erarbeitet werden, die dem Unternehmen auch ökonomische Vorteile bieten. Die Bearbeitung des Datensatzes erfolgt neben dem kNN-Algorithmus noch mit zusätzlichen Algorithmen. Dazu wird ein Entscheidungsbaum, ein Random Forest, Lineare Klassifikatoren und eine Support Vector Machine (SVM) verwendet. Anschließend erfolgt ein Vergleich dieser Algorithmen und eine Beurteilung, welcher für den Datensatz am geeignetsten ist. Abschließend werden die zuvor erarbeiteten Fragestellungen beantwortet, ehe ein finales Fazit folgt.

2. Datensatz

Der Datensatz umfasst 1.000 Kundenprofile mit insgesamt 12 Merkmalen, darunter demografische Informationen (Alter, Einkommen, Bildungsgrad). Alle Merkmale sind numerisch. Den Kunden wurden Produkte über das Telefon verkauft. Die Zielvariable wird durch die Spalte „custcat“ dargestellt und beschreibt den gewünschten Service-Umfang der Kunden. Der Service-Umfang kann mit Basic Service(1), E-Service(2), Plus Service(3) und Total Service(4) beschrieben werden.

Tabelle 1: Datensatz Analyse

Spalte	Bedeutung	“summary statistics”
region	Regionale Zuordnung des Kunden	maximum 3 minimum 1, mittelwert 2,022
tenure	Dauer der Kundenbeziehung	maximum 72 minimum 1, mittelwert 35,526
age	Alter des Kunden	maximum 77 minimum 18, mittelwert 41,684
marital	Familienstand	maximum 1 minimum 0, mittelwert 0,495
address	Verweildauer an aktueller Adresse (in Monaten)	maximum 55 minimum 0, mittelwert 11,551
income	Einkommen des Kunden	maximum 1.668.000 minimum 9.000, mittelwert 77535
ed	Bildungsniveau	maximum 5 minimum 1, mittelwert 2671
employ	Berufserfahrung	maximum 47 minimum 0, mittelwert 10,987
retire	Ruhestandsstatus	maximum 1.000 minimum 0.000, mittelwert 47

gender	Geschlecht	maximum 1 minimum 0, mittelwert 0,517
reside	Anzahl Jahre in aktueller Wohnregion	maximum 8 minimum 1, mittelwert 2,331
custcat	Kategorie des gewünschten Serviceumfangs	maximum 4 minimum 1, mittelwert 2,487

Die unterschiedlichen Bereiche der numerischen Werte könnte eine Herausforderung für das Modell darstellen, da einzelne Merkmale aufgrund ihrer Größe stärken ins Gewicht fallen können. Dies ist vor allem bei Algorithmen, welche distanzbasierte Berechnungen nutzen von großer Bedeutung, da sonst eine Verzerrung der Ergebnisse zu befürchten ist. Die begrenzte Datenmenge von lediglich 1000 Instanzen stellt eine zentrale Herausforderung für die Modellgeneralisierung dar. Insbesondere bei vier Zielklassen ist die Anzahl der Beispiele pro Klasse gering, was die Lernfähigkeit der Modelle einschränken kann. Die genaue Anzahl der einzelnen Klassen (‚custcat') wird in nachfolgender Tabelle dargestellt.

Tabelle 2: Vorkommen einzelne Klassen

Klasse	Anzahl
1	266
2	217
3	281
4	236

Diese Abbildung verdeutlicht die große Herausforderung, aufgrund der geringen Datenmenge. Anhand dieser Tabelle lässt sich die Vermutung aufstellen, dass aufgrund des Abfalls der Anzahl Klasse 2 am wenigsten gut vorhergesagt werden kann, da die wenigsten Trainingsdaten zur Verfügung stehen. Während Klasse 3 vermeintlich die beste Vorhersage liefern wird, da dort am meisten Daten zur Erkennung von Mustern zur Verfügung stehen.

Tabelle 3: Variablen pro Klasse

Custcat	Region	Tenure	Age	Martial	adress	income	ed	employ	retire	gender	reside
Klasse 1	2,09	24,68	39,66	0,42	9,39	54759	2,36	8,49	0,041	0,51	2,2
Klasse 2	1,96	43,34	41,79	0,52	12,57	74834	2,96	10,58	0,0296	0,55	2,33
Klasse 3	2	40,08	44,43	0,49	13,38	89032	2,15	14,31	0,079	0,51	2,23
Klasse 4	2,03	35,14	40,6	0,55	10,87	92000	3,37	10,23	0,0296	0,512	2,6

Eine erste Analyse der Daten beziehungsweise der Klassen mit Hilfe obiger Tabelle zeigt, dass vor allem in der Variable „income“ eine Steigerung für jeden größeren Service-Umfang ersichtlich ist. Klasse 4 zeigt mit einem Bildungsniveau (ed) von 3,37 und einer Beschäftigungsdauer von 10,23 ein vergleichsweise hohes sozioökonomisches Profil. Klasse 3 sticht durch besonders hohe Beschäftigungsdauer (14,31) hervor.

3. Fragestellung

Die übergeordnete Fragestellung, welche gleichzeitig die Aufgabenstellung beziehungsweise das Ziel der Algorithmen darstellt, lautet wie folgt: Welcher Serviceumfang wird vermutlich von dem Kunden gewünscht?

Aus dieser übergeordneten Fragestellung lassen sich unterschiedliche aufgetrennte Problem- und Zielstellungen formulieren.

Zunächst lässt sich als erstes Ziel die korrekte Vorhersage des Serviceumfangs von Kunden anhand der Daten beschreiben. Daraus kann folgende Frage formuliert werden: Kann ein neuer Kunde zuverlässig in eine Kategorie eingeordnet werden? Dafür sind die Genauigkeiten der Vorhersage der Algorithmen notwendig, um diese Frage beantworten zu können.

Ein weitere Zielstellung ist einen gezielten Vertrieb zu ermöglichen, welcher auf Basis der typischen Merkmale für Kunden mit hohem Serviceumfang umfang ermöglicht wird. Diese typischen Merkmale inklusive ihrer ungefähren Werte gilt es herauszufinden.

Einher mit dieser Zielstellung geht eine weitere, welche als Grundlage für Marketingmaßnahmen gilt. Hier sollen die Unterschiede der einzelnen Kundengruppen beziehungsweise Kunden der Service-Umfänge herausgearbeitet werden.

Die letzte Problemstellung zielt auf eine abgestimmte Kommunikation und Vertrieb nach Region sowie Geschlecht ab. Dabei sollen Unterschiede in Bezug auf die Servicewünsche zwischen Regionen und Geschlechtern erarbeitet werden. Damit soll dem Unternehmen die Möglichkeit geboten werden, durch gezielte Fokussierung auf Kommunikation und Vertrieb in bestimmten Regionen für ein bestimmtes Geschlecht mit dem höchsten Service-Umfang, finanzielle Einsparungen zu erzielen. Das heißt ein willkürlicher Vertrieb ist nicht mehr notwendig, sondern kann gezielt auf den höchsten Service-Umfang und den vermeintlich höheren Umsatz gelenkt werden.

4. Algorithmen

Bevor man mit einem Klassifikationsalgorithmus arbeitet, müssen die Daten vorbereitet werden. Der Code übernimmt dabei wichtige Schritte, damit die Modelle später korrekt und zuverlässig funktionieren.

Zuerst wird der Datensatz mit Kundendaten eingelesen und angeschaut. So bekommt man einen Überblick über die Struktur der Daten, also wie viele Zeilen und Spalten vorhanden sind und welche Informationen enthalten sind. Außerdem wird geprüft, ob es fehlende Werte gibt. Das ist wichtig, weil viele Algorithmen nicht mit leeren Feldern arbeiten können.

Danach wird der Datensatz in Merkmale (X) und Zielvariable (y) aufgeteilt. Die Merkmale beinhalten alle Informationen, die das Modell später nutzen soll, zum Beispiel demografische Daten oder Nutzungsverhalten. Die Zielvariable *custcat* gibt an, zu welcher Kundengruppe eine Person gehört. Diese Gruppenzuordnung soll das Modell vorhersagen.

Anschließend wird der Datensatz in Trainings- und Testdaten aufgeteilt. Dabei wird darauf geachtet, dass die Verteilung der Zielklassen in beiden Teilen gleichbleibt. Das sorgt für eine faire Bewertung der Modellleistung. Der Trainingsdatensatz wird verwendet, um das Modell zu trainieren. Der Testdatensatz dient später zur Überprüfung, wie gut das Modell mit neuen Daten umgehen kann.

Nach dem Training ist es sinnvoll, sich nicht nur die Genauigkeit anzuschauen, sondern auch die Evaluierungen. Die Konfusionsmatrix bietet eine Übersicht darüber, wie viele Vorhersagen korrekt oder falsch für jede Klasse getroffen wurden. Wenn auffällt, dass eine Klasse besonders häufig als eine andere vorhergesagt wird, kann das ein Hinweis darauf sein, dass sich diese Klassen in den Eingabedaten ähneln. Solche Muster sollte man festhalten und genauer betrachten. Der F1-Score pro Klasse untersucht, wie zuverlässig ein Modell bei der Vorhersage einzelner Klassen beziehungsweise der Service-Umfänge ist. Die Feature Importance gibt einen Einblick in die Bedeutung einzelner Merkmale für die Entscheidungsfindung des Modells und hilft zu verstehen, welche Variablen den größten Einfluss auf die Klassifikation haben. Die Precision vs. Recall Matrix pro Klasse visualisiert das Zusammenspiel zwischen Präzision und Sensitivität des Modells. Das heißt zwischen Genauigkeit der Vorhersage und der Fähigkeit des Modells, alle relevanten Instanzen zu erkennen. Der kombinierte

F1-Score, Precision und Recall pro Klasse erlaubt es die Stärken und Schwächen des Modells in Bezug auf die Vorhersagequalität für jede Klasse differenziert zu analysieren. Zusammenfassend ermöglichen diese Metriken tiefere und transparente Bewertungen der Modellqualität über die reine Genauigkeit hinaus.

Diese Schritte sind wichtig, damit die Modelle auf einer sauberen und gut vorbereiteten Datenbasis aufbauen. Damit können zuverlässige Ergebnisse erzielt und eventuelle Fehlerquellen besser verstanden werden.

4.1 k-Nearest-Neighbors-Algorithmus

Für die Klassifikation des gewünschten Serviceumfangs wird der kNN-Algorithmus verwendet. Dieser Algorithmus gehört zu den sogenannten nicht-parametrischen Verfahren. Das bedeutet, dass das Modell nichts „lernt“, sondern die Vorhersage erst dann trifft, wenn ein neuer Datenpunkt klassifiziert werden soll.

Die Entscheidung basiert auf den k nächsten Nachbarn im Datensatz. Dabei wird geschaut, welche Klassen diese Nachbarn haben, und die häufigste Klasse wird dann als Vorhersage genommen. Standardmäßig verwendet scikit-learn dabei die euklidische Distanz, also den Luftlinienabstand im Merkmalsraum.[1]

Wichtig bei kNN ist die Wahl des Parameters k, also wie viele Nachbarn berücksichtigt werden sollen. Dieser Parameter zählt zu den sogenannten Hyperparametern, da er nicht aus den Daten gelernt wird, sondern vor dem Training festgelegt werden muss. Ein zu kleiner k-Wert kann zu Overfitting führen, während ein zu großer k-Wert das Modell zu stark glättet (Underfitting).[2]

Um den optimalen Wert zu finden, wird ein Hyperparameter-Tuning mithilfe einer GridSearch über k=1 bis k=20 durchgeführt. Für jeden Wert wird die mittlere Genauigkeit anhand einer 5-fachen Kreuzvalidierung berechnet.[3] Zusätzlich wird die Modellgüte für jedes k auf dem abgetrennten Testdatensatz evaluiert, um die Generalisierungsfähigkeit auf unbekannte Daten zu bewerten.

Dabei zeigte sich, dass der beste Wert der Kreuzvalidierung bei k=17 mit einer durchschnittlichen Genauigkeit von 0.393 lag, während das Modell auf dem Testset mit k=6 die höchste Genauigkeit von 0.400 erreichte. Da das Ziel der Modellierung in einer

[1] (scikit-learn, o. J.-c)
[2] (Höchenberger, 2025)
[3] (Arnold et al., 2024, S.844-845)

möglichst zuverlässigen Klassifikation auf neuen, realen Daten besteht, wird bewusst der Testset-optimale Wert k=6 gewählt.

Tabelle 4: Vergleich von Kreuzvalidierung und Testdaten im kNN-Modell

Kriterium	**Kreuzvalidierung (CV)**	**Testset-Vergleich**
Datengrundlage	Nur Trainingsdaten, mehrfach unterteilt	Vollständig abgetrennter Testdatensatz
Ziel	Möglichst stabiles Ergebnis im Schnitt	Gute Leistung auf unbekannte Daten
Ergebnis	k=17	k=6
Vorteil	Weniger anfällig für Zufall oder Ausreißer	Realitätsnahe Prüfung auf echten neuen Daten
Nachteil	Keine echte Generalisierungsprüfung	Ergebnis kann zufällig verzerrt sein
Empfehlung	Gut für theoretische Bewertung	Gut für Praxis, da näher an der späteren Praxis

Die finale Entscheidung fällt damit, auf den Wert mit dem besten Praxiserfolg, also k=6. Die Ergebnisse beider Verfahren werden zudem unten grafisch dargestellt, um die Entwicklung der Modellgüte in Abhängigkeit vom k-Wert zu visualisieren. Diese Vorgehensweise stellt sicher, dass die Auswahl nicht willkürlich erfolgt, sondern unter Berücksichtigung des Ziels zur hohen Vorhersagequalität neuer Daten.

Fehler! Verweisquelle konnte nicht gefunden werden. zeigt den Verlauf der Modellgenauigkeit (Accuracy) auf dem Trainings- und Testdatensatz in Abhängigkeit vom gewählten k-Wert (1 bis 20).

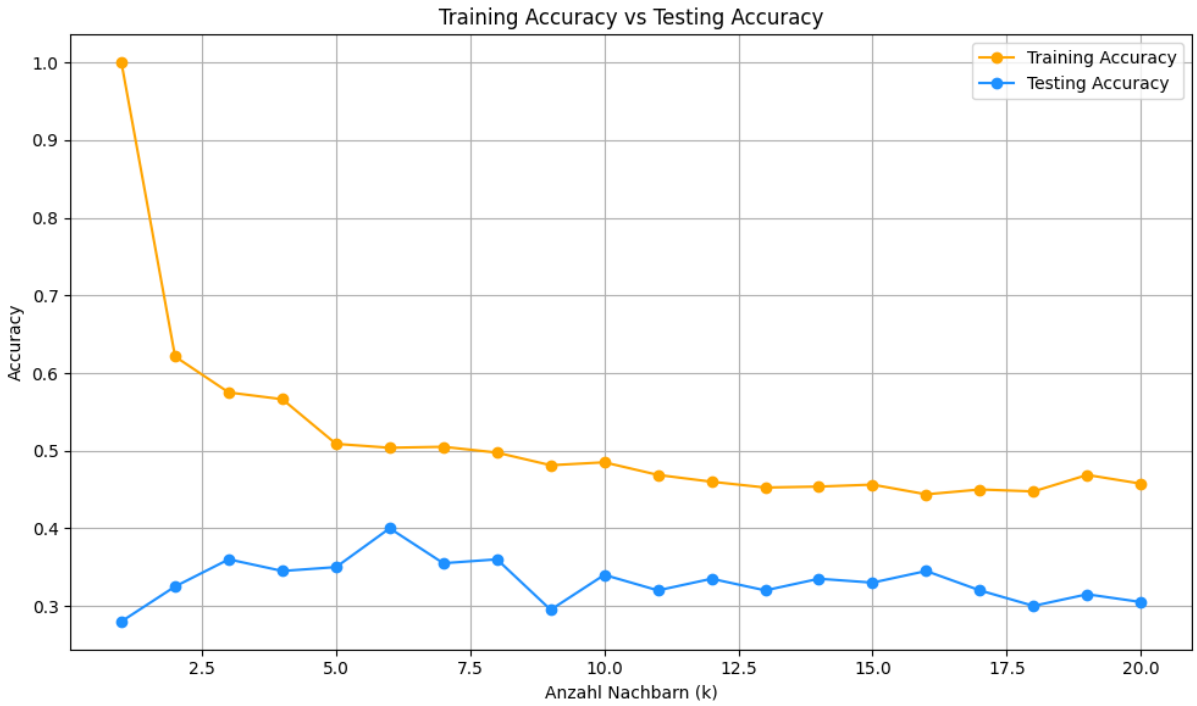

Abbildung 1: Trainings- und Testgenauigkeit in Abhängigkeit von k

Deutlich erkennbar ist, dass bei sehr kleinen k-Werten eine starke Überanpassung (Overfitting) auftritt. Die Trainingsgenauigkeit ist hier nahezu perfekt, während die Testgenauigkeit deutlich niedriger ausfällt. Bei zunehmendem k sinkt die Trainingsgenauigkeit zwar ab, die Testgenauigkeit hingegen stabilisiert sich und erreicht ihren Höchstwert bei k = 6. In diesem Bereich liegt ein gutes Gleichgewicht zwischen Overfitting und Underfitting vor. Daher wird k = 6 als optimaler Wert ausgewählt.

Abbildung 2 die durchschnittliche Modellgüte (Accuracy) auf Basis einer 5-fachen Kreuzvalidierung für verschiedene k-Werte im Bereich von 1 bis 20.

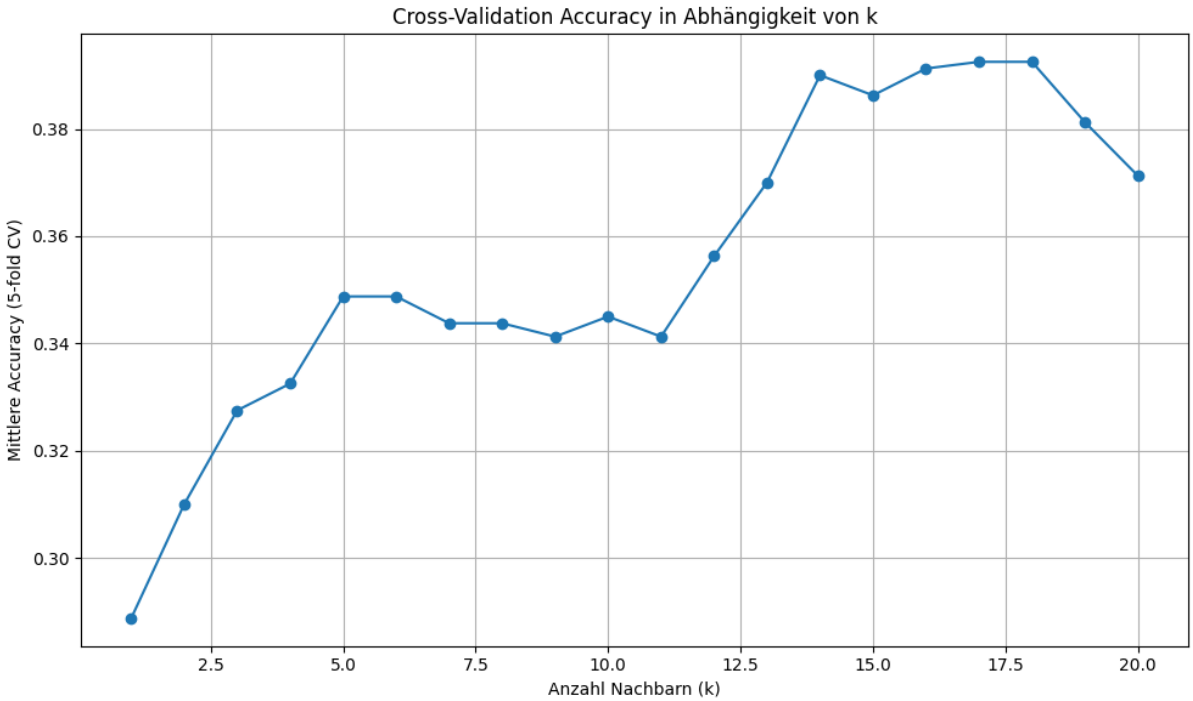

Abbildung 2: Kreuzvalidierungsgenauigkeit in Abhängigkeit von k

Zu erkennen ist ein kontinuierlicher Anstieg der mittleren Genauigkeit mit zunehmendem k, wobei der Höchstwert bei k = 17 erreicht wird. Dieser Punkt stellt aus Validierungssicht den optimalen Wert dar. Dieser liefert die höchste durchschnittliche Vorhersagequalität auf den unterteilten Trainingsdaten.

Dennoch wird in dieser Arbeit $k = 6$ verwendet, da dieses Modell auf dem vollständig abgetrennten Testdatensatz die beste Leistung zeigt. Die Wahl orientiert sich somit bewusst an der realen Einsatzsituation und nicht an der theoretisch stabileren Kreuzvalidierung.

Zur Einordnung und zum Vergleich wird in Abbildung 3 die Genauigkeit auf Trainingsdaten, Testdaten und im Mittel der Cross-Validation gegenübergestellt.

```
Cross-Validation Accuracy (Mittelwert): 0.3488
Genauigkeit auf Trainingsdaten: 0.5038
Genauigkeit auf Testdaten: 0.4000
```

Abbildung 3: Vergleich der Modellgenauigkeit auf Trainingsdaten, Testdaten und in der Kreuzvalidierung bei kNN

Diese Übersicht macht deutlich, wie stark sich die Modellgüte je nach Datengrundlage unterscheiden kann und warum eine bewusste Entscheidung für die Testdatenleistung getroffen wurde. Auffällig ist der große Unterschied zwischen Trainings- und Testgenauigkeit, was auf eine mangelnde Generalisierungsfähigkeit des Modells hindeutet. Dies deutet entweder auf Overfitting (Überanpassung an die Trainingsdaten) oder

Underfitting (unzureichende Modellkomplexität) hin. Es zeigt also, dass das Modell auf dem Testdatensatz kurzfristig gut abschneidet, aber insgesamt noch Optimierungspotenzial besteht.

Zur Bewertung der Modellleistung wird zunächst die Gesamttreffergenauigkeit (Accuracy) berechnet, die angibt, welcher Anteil der Testbeispiele korrekt klassifiziert wurde. Im Anschluss wird eine Konfusionsmatrix erstellt (vgl. Abbildung 4), die eine detaillierte Übersicht über korrekte und fehlerhafte Klassifikationen je Klasse liefert. Dadurch lässt sich erkennen, in welchen Bereichen das Modell zuverlässig arbeitet und in welchen es zu Verwechslungen kommt.

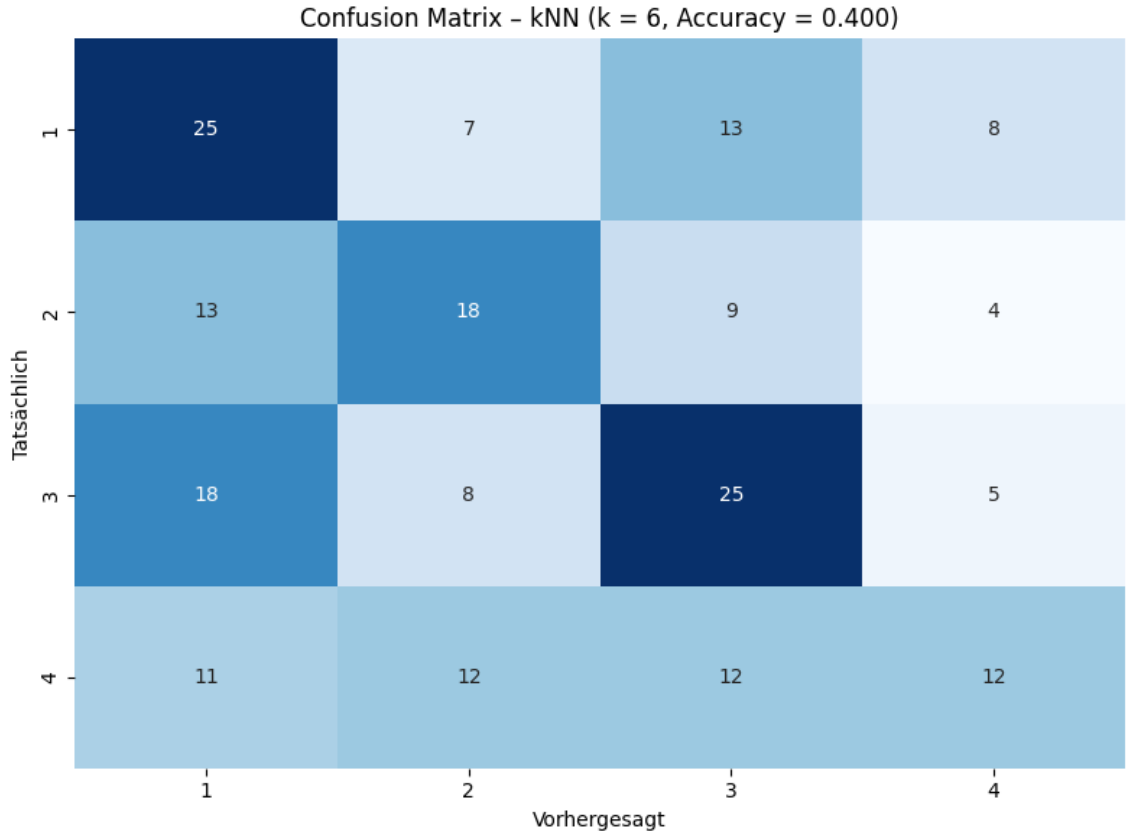

Abbildung 4: Konfusionsmatrix des kNN-Modells mit

Die Konfusionsmatrix zeigt die Klassifikationsergebnisse des finalen kNN-Modells. Richtig klassifizierte Instanzen befinden sich auf der Diagonalen, während die übrigen Felder Fehlklassifikationen sind. Auffällig ist, dass bestimmte Klassen wie 1 und 3 vergleichsweise oft korrekt erkannt werden, während andere häufiger verwechselt werden. Das Modell erreicht eine Genauigkeit von 0.400. Das Ergebnis weist auf eine begrenzte Eindeutigkeit zwischen den Klassen hin.

Um die Modellleistung noch genauer zu bewerten, wird ein Classification Report erstellt (vgl. Abbildung 5). Dieser enthält wichtige Kennzahlen wie Precision, Recall und den F1-Score für jede Klasse.

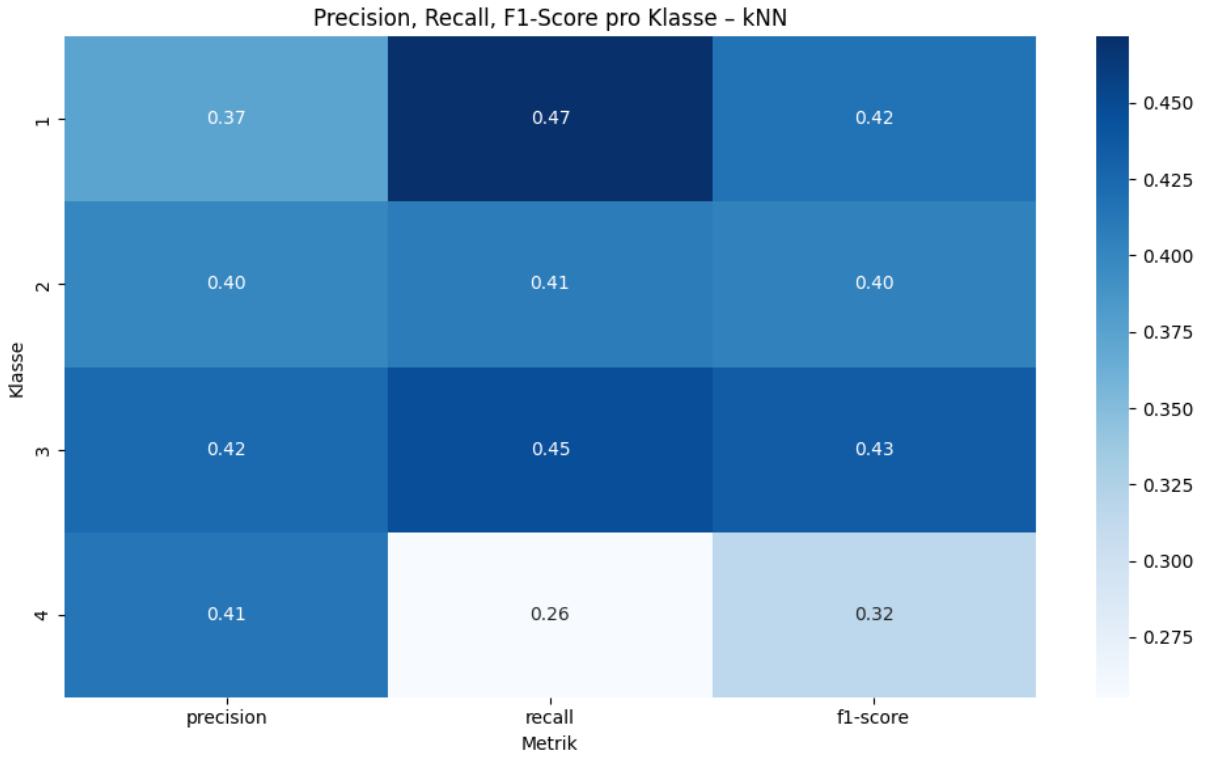

Abbildung 5: Precision, Recall und F1-Score pro Klasse des kNN-Modells

Die einzelnen Zeilen repräsentieren die einzelnen vier Klassen, während die Spalten die Bewertungsmetriken darstellen. Die Precision gibt an, wie zuverlässig eine vorhergesagte Klasse tatsächlich korrekt ist. Der Recall beschreibt, wie gut das Modell eine Klasse erkennt, wenn sie tatsächlich vorliegt, also den Anteil korrekt erkannter Instanzen an allen tatsächlichen Vorkommen der Klasse. Der F1-Score bildet das harmonische Mittel aus Precision und Recall und stellt eine zusammenfassende Bewertung der Modellgüte pro Klasse dar.

Auffällig ist, dass Klasse 3 in allen drei Metriken besonders gute Werte erreicht und somit am zuverlässigsten vom Modell erkannt wird. Auch Klasse 1 weist einen hohen Recall-Wert auf, was bedeutet, dass sie häufig erkannt wird, wenn sie tatsächlich vorliegt. Klasse 4 hingegen zeigt mit einem Recall von nur 0.26 deutliche Schwächen in der Erkennung, obwohl die Precision mit 0.41 durchaus solide ausfällt. Das deutet darauf hin, dass Klasse 4 zwar selten erkannt wird, aber wenn, dann meist korrekt. Klasse 2 zeigt in allen Metriken mittelmäßige Werte um die 0.40, ohne größere Ausreißer.

Die Modellleistung pro Klasse liegt damit zwischen 0.26 und 0.47. Das verdeutlicht, dass das Modell in der Lage ist, einige Klassen relativ zuverlässig zu erkennen, während es bei anderen noch deutliches Verbesserungspotenzial gibt. Die Visualisierung hilft dabei, die Stärken und Schwächen des Modells auf Klassenebene besser zu verstehen.

Im nächsten Schritt wird überprüft, welche Merkmale am wichtigsten für die Vorhersage sind. Abbildung 6 zeigt die Ergebnisse der Feature-Auswahl mittels ANOVA F-Test. Diese Methode bewertet jedes Merkmal danach, wie stark es mit der Zielvariable (custcat) zusammenhängt.

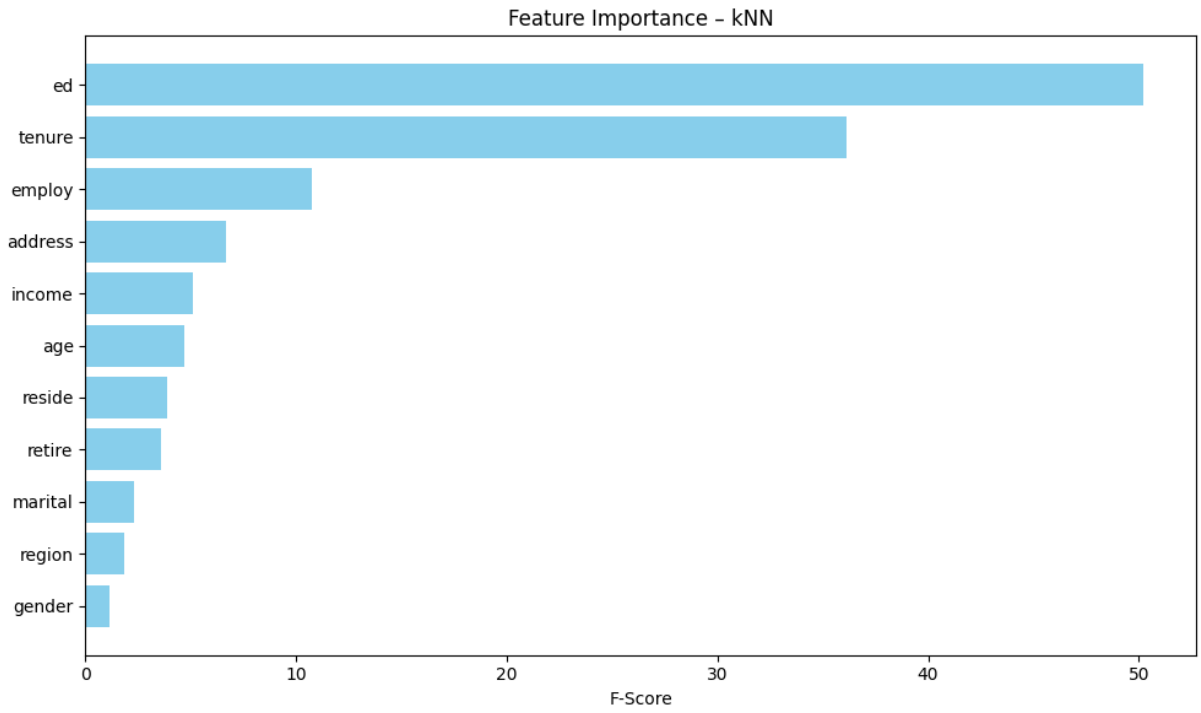

Abbildung 6: Merkmalswichtigkeit im kNN-Modell

Je höher der F-Score, desto stärker unterscheidet sich die Verteilung des Merkmals zwischen den Klassen und desto größer ist dessen Aussagekraft für das Modell. Auffällig ist, dass die Merkmale „ed“ (Bildungsniveau) und „tenure“ (Dauer der Kundenbeziehung) mit Abstand die höchsten F-Scores aufweisen. Dies deutet darauf hin, dass sie den größten Beitrag zur Klassifikation des Serviceumfangs leisten. Im Gegensatz dazu zeigen einige andere Merkmale nur geringe Eindeutigkeiten und somit wenig Einfluss auf das Modell. Diese Analyse liefert wichtige Hinweise darauf, welche Variablen in der Entscheidungsfindung des kNN-Algorithmus eine wichtige Rolle spielen. Diese Erkenntnis kann auch für eine gezielte Feature-Reduktion in zukünftigen

Modellversionen genutzt werden. Als letzte Abbildung rundet sie die Evaluation des kNN-Modells ab und unterstützt die Interpretation der vorherigen Klassifikationsergebnisse.

4.2 Entscheidungsbaum

Für Entscheidungsbäume gibt es unterschiedliche Algorithmen, auf deren Grundlage der Aufbau des Baumes basiert. In scikit-learn wird ein optimierter Classifiaction and Regression Tree (CART) Algorithmus verwendet.[4] CART erstellt einen Baum der aus Knoten, Zweigen und Blättern besteht. Knoten entsprechen den Entscheidungspunkten, während die Blätter die Ergebnisse der Entscheidungen darstellen, abschließend sagen die Blätter eine Klasse vorher. Um aus dem Datensatz einen Entscheidungsbaum zu erzeugen, werden die daten rekursiv gesplittet. Mit dem Ziel möglichst homogene Klassen in jedem Blatt zu haben. Dafür verwendet CART die Gini-Impurity als Maß für die Reinheit in jedem Knoten. Sie beschreibt wie gemischt die Klassen in einem Knoten sind. Je niedriger die Gini-Impurtiy, desto reiner der Knoten. Der Algrotihmus sucht bei jedem Split nach dem Schwellenwert, der die Gini-Impurity am stärksten reduziert und wendet diesen Split an. So entsteht ein Entscheidungsbaum, der möglichst klare und zuverlässig Vorhersagen trifft.[5] Scikit ermöglicht darüber hinaus neben der Verwendung von Gini als Kriterium auch entropy und log_loss.[6] Da für den hier optimierten Entscheidungsbaum Gini verwendet wird, werden die anderen Kriterien im Rahmen dieser Dokumentation nicht weitergehend betrachtet.

Um einen möglichst optimalen Entscheidungsbaum zu erstellen, wird dieser mit Hilfe von GridSearchCV optimiert. Die Entscheidung auf GridSearchCV anstelle von RandomSearch viel aufgrund mehrerer Faktoren. Während GridSearchCV alle Hyperparameter in dem Bereich testet, beschränkt sich RandomSearch nur auf eine geringere Anzahl. Daher ist die Wahrscheinlichkeit mit RandomSearch, die optimalen Parameter zu finden geringer. Da der Datensatz mit 1000 Einträgen vergleichsweise klein ist, beansprucht GridSearchCV nicht übermäßig viel Zeit, weshalb diese eingesetzt wird, um die optimalen Parameter herauszufinden.[7]

Zur Erarbeitung der optimalen Parameter wird ein iterativer GridSearchCV-Ansatz verwendet. Eine Anpassung ergibt keine Veränderung in den Genauigkeiten. Der Entscheidungsbaum zeigt eine stabile, aber leicht unterfittende Tendenz. Die etwas höhere Trainingsgenauigkeit und ausgeglichene Testgenauigkeit und Cross-Validation-Genauigkeit deutet auf ein reguliertes Modell mit begrenzter Komplexität. Das Modell

[4] (scikit-learn developers, o. J.-a)
[5] (Lewis, 2000, S.2-5)
[6] (scikit-learn developers, o. J.-b)
[7] (Shams et al., 2024, S. 35319); (Taufiq et al., 2024, S.181)

zeigt zwar keine Überanpassung, aber auch keine volle Ausschöpfung des Potenzials der Daten.

Tabelle 5: Iterative GridSearch - Entscheidungsbaum

Cross-Validation Accuracy (Mittelwert): 0.3812 Genauigkeit auf Trainingsdaten: 0.4587 Genauigkeit auf Testdaten: 0.3850	param_grid = { 'max_depth': range(1,10,1), 'min_samples_leaf': range(1, 20, 1), 'min_samples_split': range(2, 20, 2), ‘criterion’: [“entropy”, “gini”, “log_loss”]}
Cross-Validation Accuracy (Mittelwert): 0.3812 Genauigkeit auf Trainingsdaten: 0.4587 Genauigkeit auf Testdaten: 0.3850	param_grid ={ 'max_depth': range(3,6,1), 'min_samples_leaf': range(4, 10, 1), 'min_samples_split': range(2, 10, 2), 'criterion': ["entropy", "gini", "log_loss"], 'splitter': ["best"]}
Cross-Validation Accuracy (Mittelwert): 0.3812 Genauigkeit auf Trainingsdaten: 0.4587 Genauigkeit auf Testdaten: 0.3850	param_grid = { 'max_depth': range(3,6,1), 'min_samples_leaf': range(6, 12, 1), 'min_samples_split': range(2, 10, 2)}
Cross-Validation Accuracy (Mittelwert): 0.3812 Genauigkeit auf Trainingsdaten: 0.4587 Genauigkeit auf Testdaten: 0.3850	param_grid = { 'max_depth': range(4,9), 'min_samples_leaf': range(3, 8), 'min_samples_split': range(2, 6) }
Cross-Validation Accuracy (Mittelwert): 0.3800 Genauigkeit auf Trainingsdaten: 0.4587 Genauigkeit auf Testdaten: 0.3850	param_grid = { 'max_depth': range(4,11), 'min_samples_leaf': range(3, 7),

	'min_samples_split': [2,3,4] }

Folgende Tabelle stellt die optimalen Parameter dar:

Tabelle 6: Parameter - Entscheidungsbaum

Parameter	Wert
Max_depth	4
Min_samples_leaf	6
Min_samples_split	2

Folgende Darstellung stellt den Entscheidungsbaum dar:

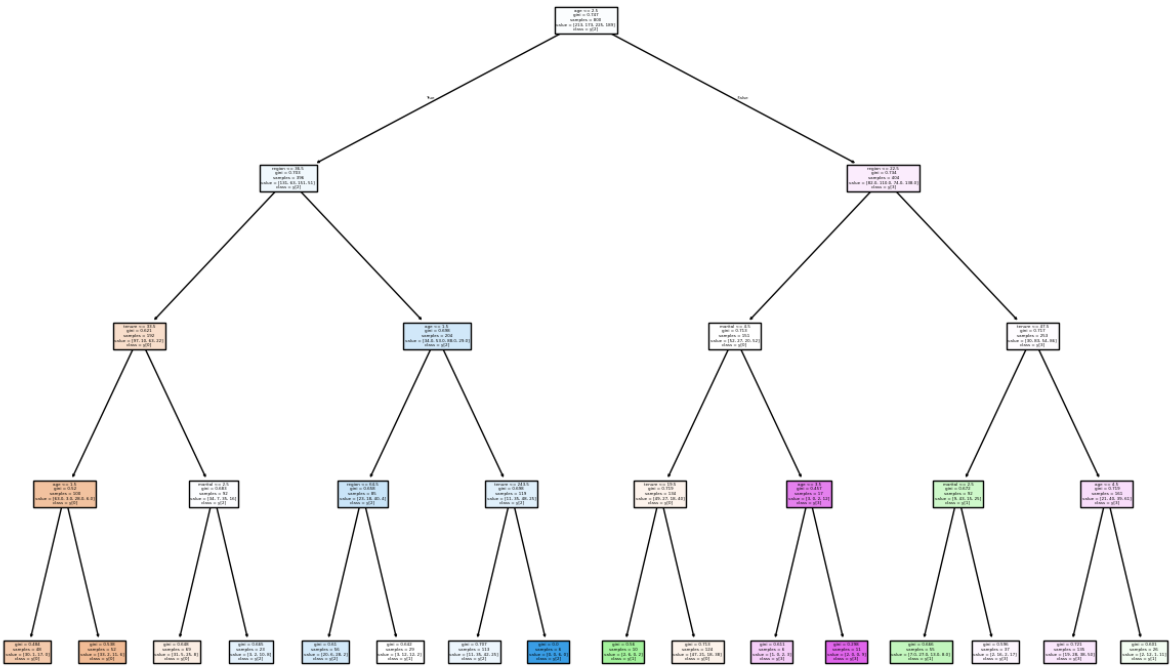

Abbildung 7: Darstellung Entscheidungsbaum

Die Evaluierung des Entscheidungsbaum-Modells zeigt, dass die Leistung insgesamt mäßig ist. Die Trainingsgenauigkeit weist auf ein begrenztes Lernvermögen des Modells hin. Die ähnliche Testgenauigkeit ist akzeptabel. Insgesamt kann das Modell als leicht underfittet, angesehen werden. Die Cross-Validation-Genauigkeit von 38%,

welche nahe an der Testgenauigkeit liegt, spricht für eine verlässliche Generalisierungsfähigkeit.

Die geringe Differenz zwischen Trainings- und Testgenauigkeit deutet zwar auf kein starkes Underfitting hin, jedoch ist die Gesamtleistung zu niedrig, um das Modell als zuverlässig oder einsatzbereit zu bewerten. Aufgrund der niedrigen Different und da der Entscheidungsbaum in dieser Dokumentation als Ergänzung gilt, wird auf eine tiefere Feinabstimmung des Entscheidungsbaums verzichtet. Eine einmalige Optimierung mittels GridSearchCV schafft demnach die Basis für weitere Analysen.

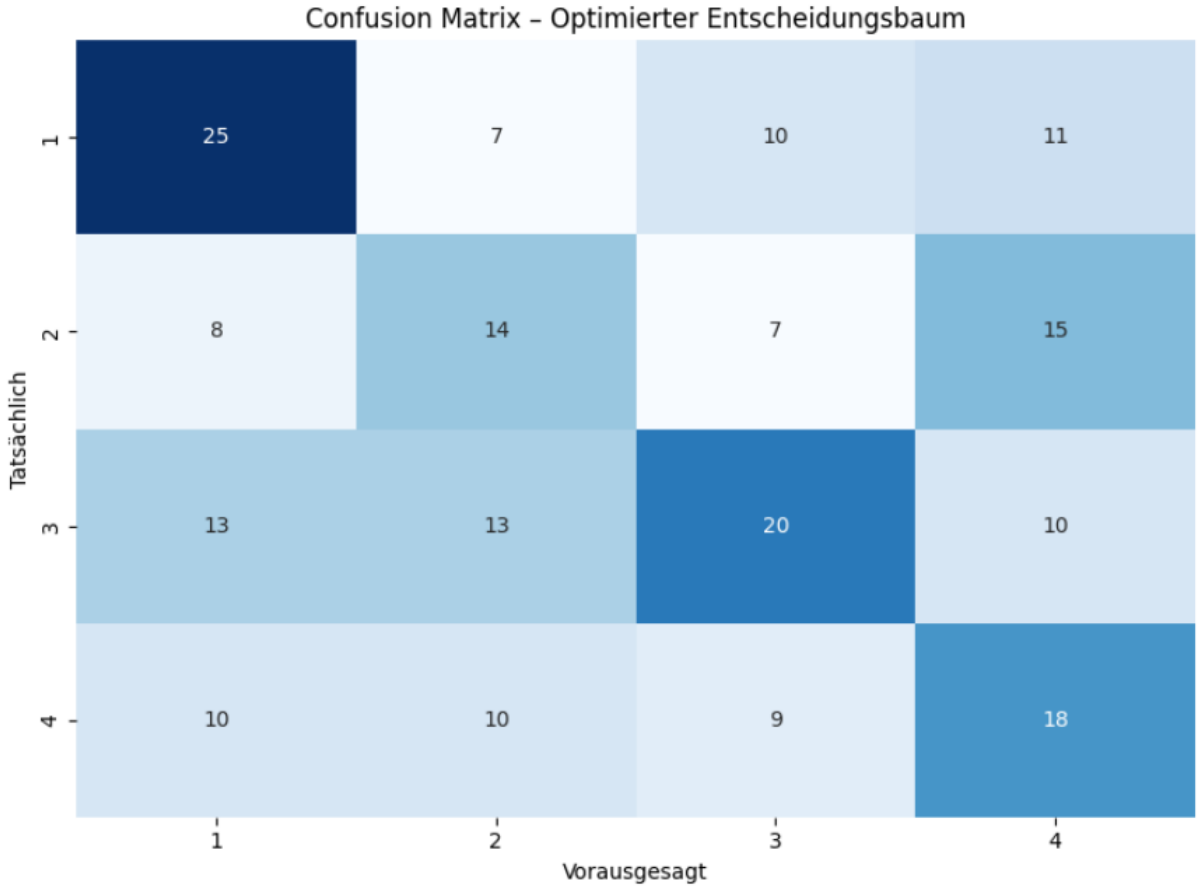

Abbildung 8: Confusion-Matrix - Entscheidungsbaum

Genauigkeit: 0,385

Insgesamt kann die Genauigkeit im Vergleich zu einem nicht optimierten Entscheidungsbaum um 0,02 gesteigert werden.

Die Abbildung zeigt, dass der Service-Umfang von 1 am häufigsten korrekt vorhergesagt werden kann. Des Weiteren wird ein Service-Umfang von 2 am häufigsten als 4 vorausgesagt.

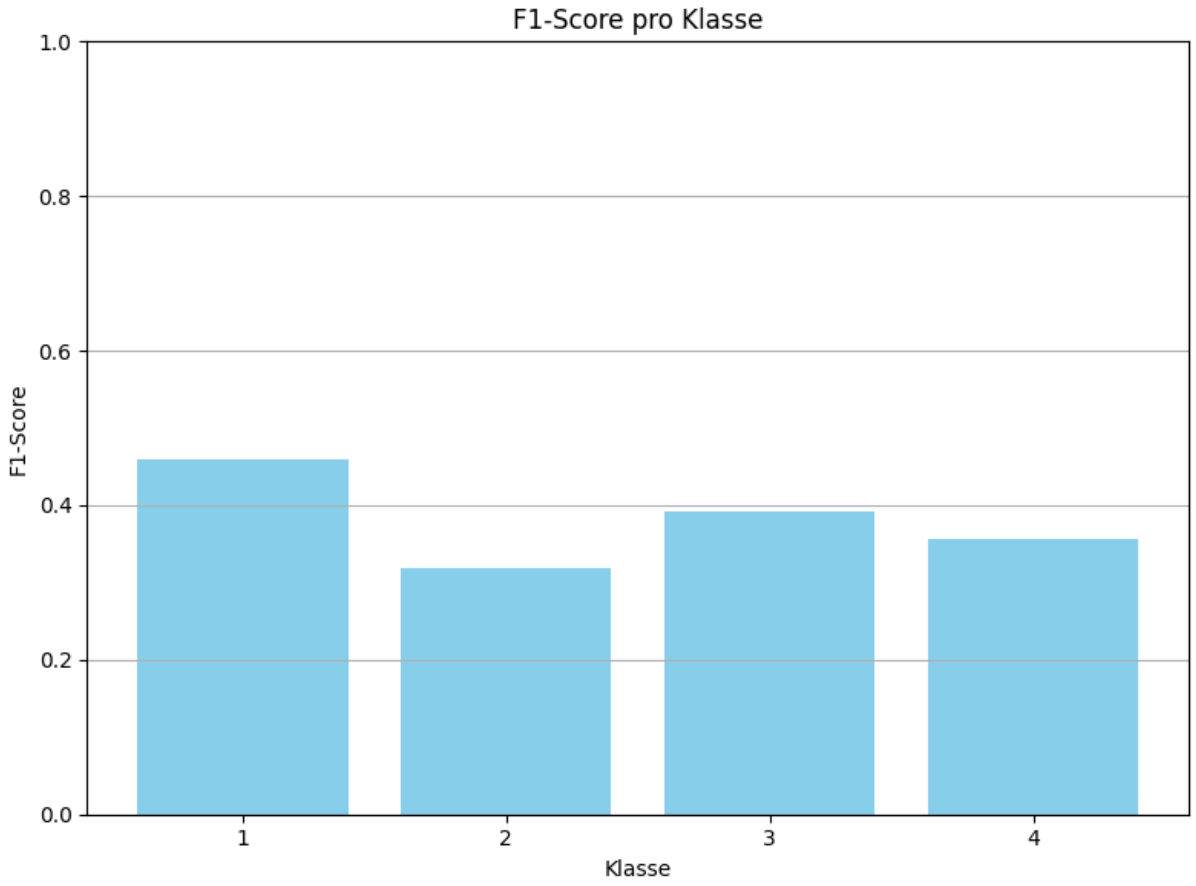

Abbildung 9: F1-Score pro Klasse – Entscheidungsbaum

Die Betrachtung der F1-Scores zeigt Scores im Bereich von etwa 30% bis zu 45%. Dies lässt darauf schließen, dass der Entscheidungsbaum in seiner aktuellen Form nicht optimal ist. Der Verdacht aus der Analyse der Confusion-Matrix bestätigt sich durch den F1-Score und zeigt erneut, dass vor allem Klasse 2 schlecht vorhergesagt wird.

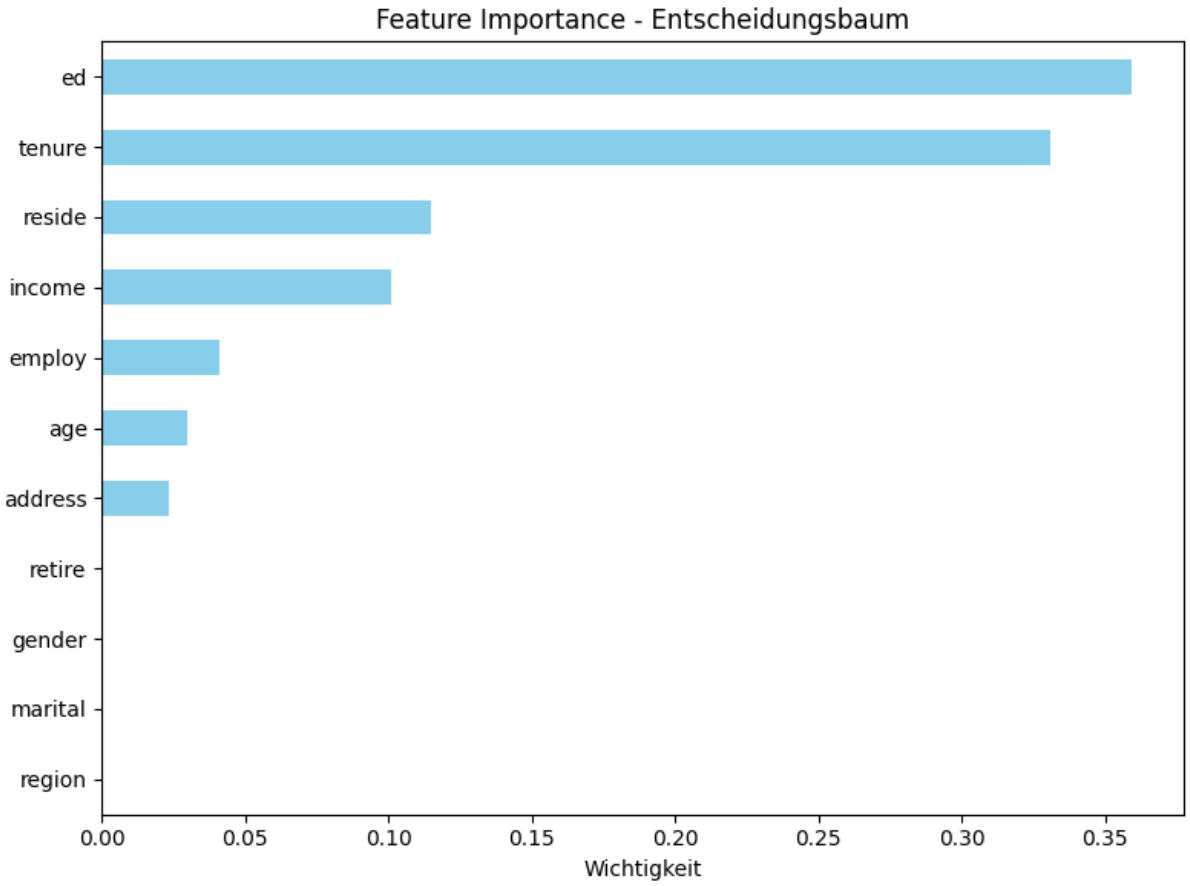

Abbildung 10: Feature Importance – Entscheidungsbaum

Eine Wichtigkeit von über 0,1 erreichen die Kriterien: Ed, Tenure, reside und income. Vor allem ed und tenure sind mit über 30% sehr wichtige Features für die Vorhersage des Entscheidungsbaumes. Gleichzeitig besitzen gender, retire und martial keinerlei Bedeutung für die Vorhersage.

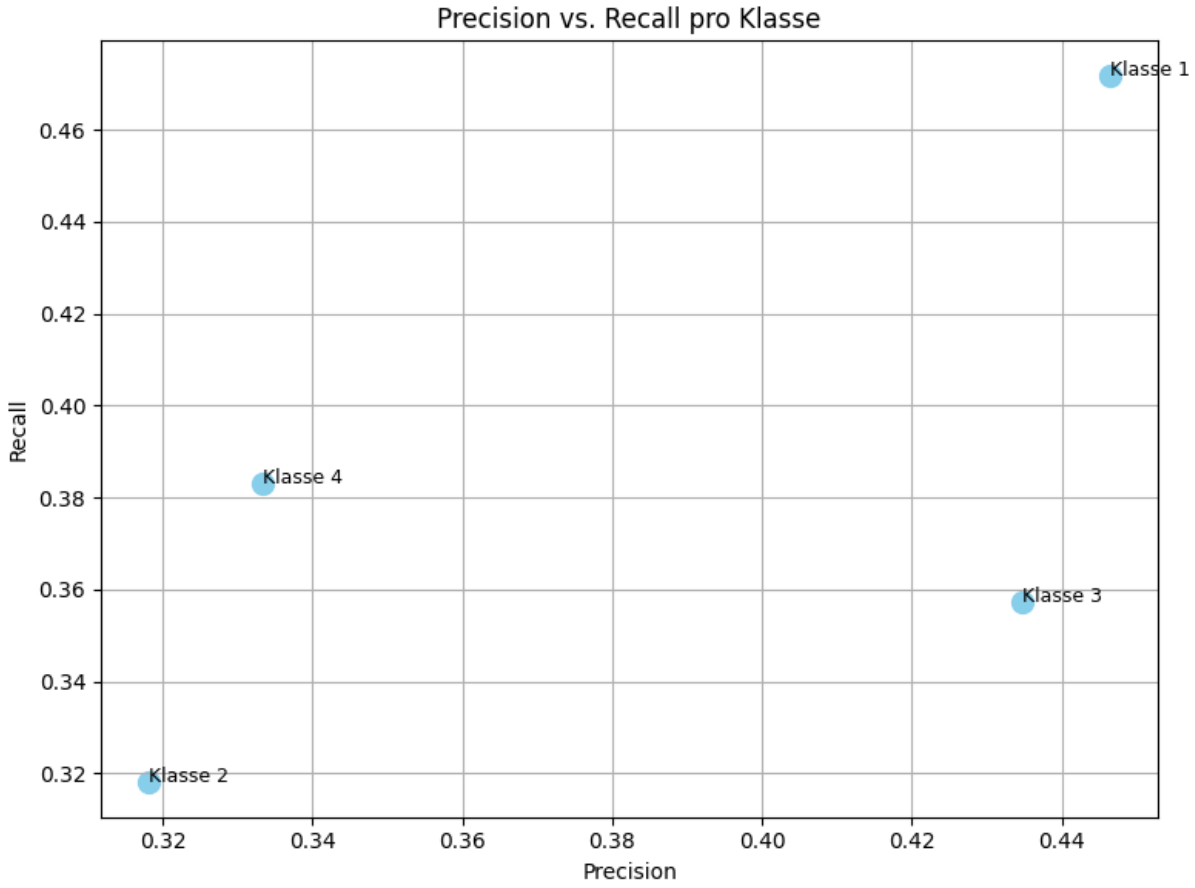

Abbildung 11: Precision vs. Recall – Entscheidungsbaum

Die Precision vs. Recall Darstellung zeigt eine deutliche Leistungsdifferenz zwischen den Klassen. Während Klasse 1 am zuverlässigsten klassifiziert wird, weist Klasse 2 die schwächste Modellleistung auf. Klasse 3 und 4 zeigen ein asymmetrisches Verhalten. Klasse 3 wird zwar mit hoher Präzision erkannt, aber gleichzeitig auch mit geringer Sensitivität. Dahingegen wird Klasse 4 häufiger erkannt, aber weniger präzise vorhergesagt. Diese Unterschiede deuten auf eine ungleichmäßige Trennschärfe des Modells hin.

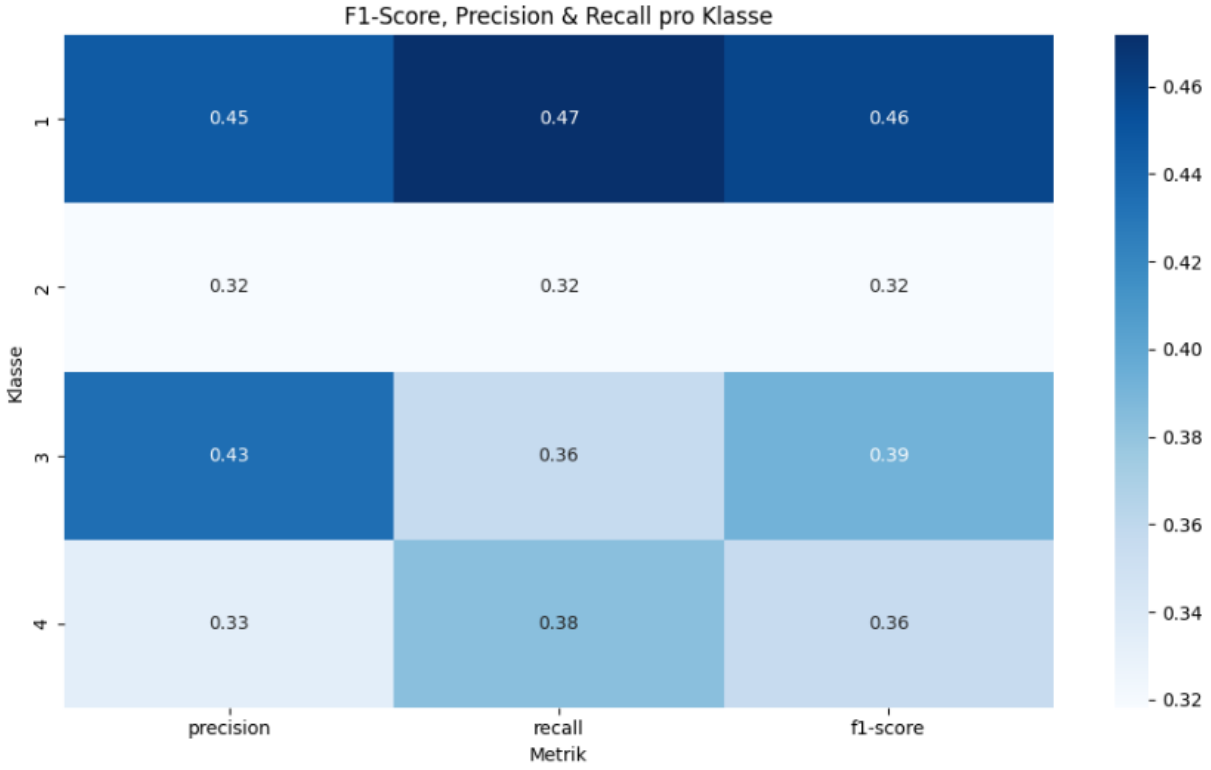

Abbildung 12: F1-Score, Precison & Recall – Entscheidungsbaum

Die Heatmap zur Modellleistung zeigt eine deutliche Heterogenität zwischen den Klassen. Klasse 1 weist mit einem F1-Score von 0.46 die höchste Klassifikationsgüte auf, während Klasse 2 mit einem Wert von 0.32 deutlich abfällt.

Obwohl das Modell mittels GridSearchCV optimiert wurde, zeigen die Leistungskennzahlen, dass die Klassifikation insbesondere für Klasse 2 unzureichend ist. Dies legt nahe, dass entweder die Hyperparameter-Suchräume nicht ausreichend gewählt wurden oder die zugrunde liegende Datenstrukturen die Modellleistung limitieren. Ein zusätzlicher Durchlauf mit weitreichenden Parameterräumen wurde durchgeführt, dennoch wurden dieselben Parameter gefunden. Die beobachtet Schwächen in der Klassifikation sind daher weniger auf suboptimale Modellparameter als vielmehr auf den eingeschränkten Datensatz zurückführen.

4.3 Random Forest

Zur Optimierung des Random-Forest-Modells wird ebenfalls ein iterative Grid-Search-Ansatz gewählt. Die Parameterbereiche werden schrittweise angepasst, basierend auf der beobachteten Diskrepanz zwischen Trainings- und Testgenauigkeit sowie der mittleren Kreuzvalidierungsgenauigkeit. Ziel ist es, ein Modell zu identifizieren, das ein möglichst geringe Überanpassung zeigt und gleichzeitig eine stabile Generalisierungsleistung erzielt.

Nachfolgende Tabelle zeigt die grobe Annäherung an:

Tabelle 7: Iterative GirdSearch - Random Forest

Cross-Validation Accuracy (Mittelwert): 0.3800 Genauigkeit auf Trainingsdaten: 0.9463 Genauigkeit auf Testdaten: 0.3950	param_grid = { 'n_estimators': [110, 115], 'max_features': ["sqrt", "log2", None], 'max_depth': [None, 10, 15], 'min_samples_split': [3, 5], 'min_samples_leaf': [1, 2], 'bootstrap': [True, False] }
Cross-Validation Accuracy (Mittelwert): 0.3900 Genauigkeit auf Trainingsdaten: 0.9413 Genauigkeit auf Testdaten: 0.3900	param_grid = { 'n_estimators': [115, 125], 'max_features': ["sqrt", "log2", None], 'max_depth': [None, 10, 12], 'min_samples_split': [4, 6], 'min_samples_leaf': [2, 3], 'bootstrap': [True, False] }
Cross-Validation Accuracy (Mittelwert): 0.3900 Genauigkeit auf Trainingsdaten: 0.9413 Genauigkeit auf Testdaten: 0.3900	param_grid = { 'n_estimators': [125, 135], 'max_depth': [None, 12, 14], 'min_samples_split': [3, 5],

	'min_samples_leaf': [3, 4], 'bootstrap': [True, False] }
Cross-Validation Accuracy (Mittelwert): 0.3900 Genauigkeit auf Trainingsdaten: 0.9413 Genauigkeit auf Testdaten: 0.3900	param_grid = { 'n_estimators': [125, 135], 'max_depth': [None, 12, 14], 'min_samples_split': [3, 5], 'min_samples_leaf': [1, 3], 'bootstrap': [True, False] }

Cross-Validation Accuracy (Mittelwert): 0.3775
Genauigkeit auf Trainingsdaten: 0.9413
Genauigkeit auf Testdaten: 0.3850
	param_grid = {
	'n_estimators': [120, 130],
	'max_depth': [None, 11, 14],
	'min_samples_split': [3, 4],
	'min_samples_leaf': [1, 3],
	'bootstrap': [True, False]
	}

Cross-Validation Accuracy (Mittelwert): 0.3900
Genauigkeit auf Trainingsdaten: 0.9413
Genauigkeit auf Testdaten: 0.3900
	param_grid = {
	'n_estimators': [125, 130],
	'max_depth': [None, 12, 14],
	'min_samples_split': [3, 5],
	'min_samples_leaf': [1, 3],
	'bootstrap': [True, False]
	}

Cross-Validation Accuracy (Mittelwert): 0.3900 Genauigkeit auf Trainingsdaten: 0.9413 Genauigkeit auf Testdaten: 0.3900	param_grid = { 'n_estimators': [125, 130], 'max_features': ["sqrt", "log2", None], 'max_depth': [None, 12, 14],

	'min_samples_split': [4, 6], 'min_samples_leaf': [2, 3], 'bootstrap': [True, False] }

Zur Optimierung und Bewertung der Modelleistung wurde eine 5-fache Kreuzvalidierung eingesetzt. Diese wurde sowohl im Rahmen der Hyperparameteroptimierung mittels GridSearchCV als auch zur unabhängigen Validierung des finalen Modells verwendet. Die wiederholte Anpassung der Hyperparameter führte zu keiner signifikanten Verbesserung der Generalisierungsleistung. Die konstant hohe Trainingsgenauigkeit (>94%) bei gleichzeitiger niedriger Testgenauigkeit (ca. 39%) und mittlerer Kreuzvalidierungsgenauigkeit (ca. 39%) spricht für strukturelles Overfitting. Daraus lässt sich schließen, dass Random Forest in diesem Kontext nicht das geeignetste Modell ist. Dennoch werden nachfolgend die Ergebnisse des Modells genauer betrachtet. Insgesamt konnte die Genauigkeit gegenüber eines nicht optimierten Random Forests um 0,02 gesteigert werden. Nachfolgend sind die Genauigkeiten des finalen Modells einzusehen:

```
Cross-Validation Accuracy (Mittelwert): 0.3900
Genauigkeit auf Trainingsdaten: 0.9413
Genauigkeit auf Testdaten: 0.3900
```

Abbildung 13: Genauigkeiten - Random Forest

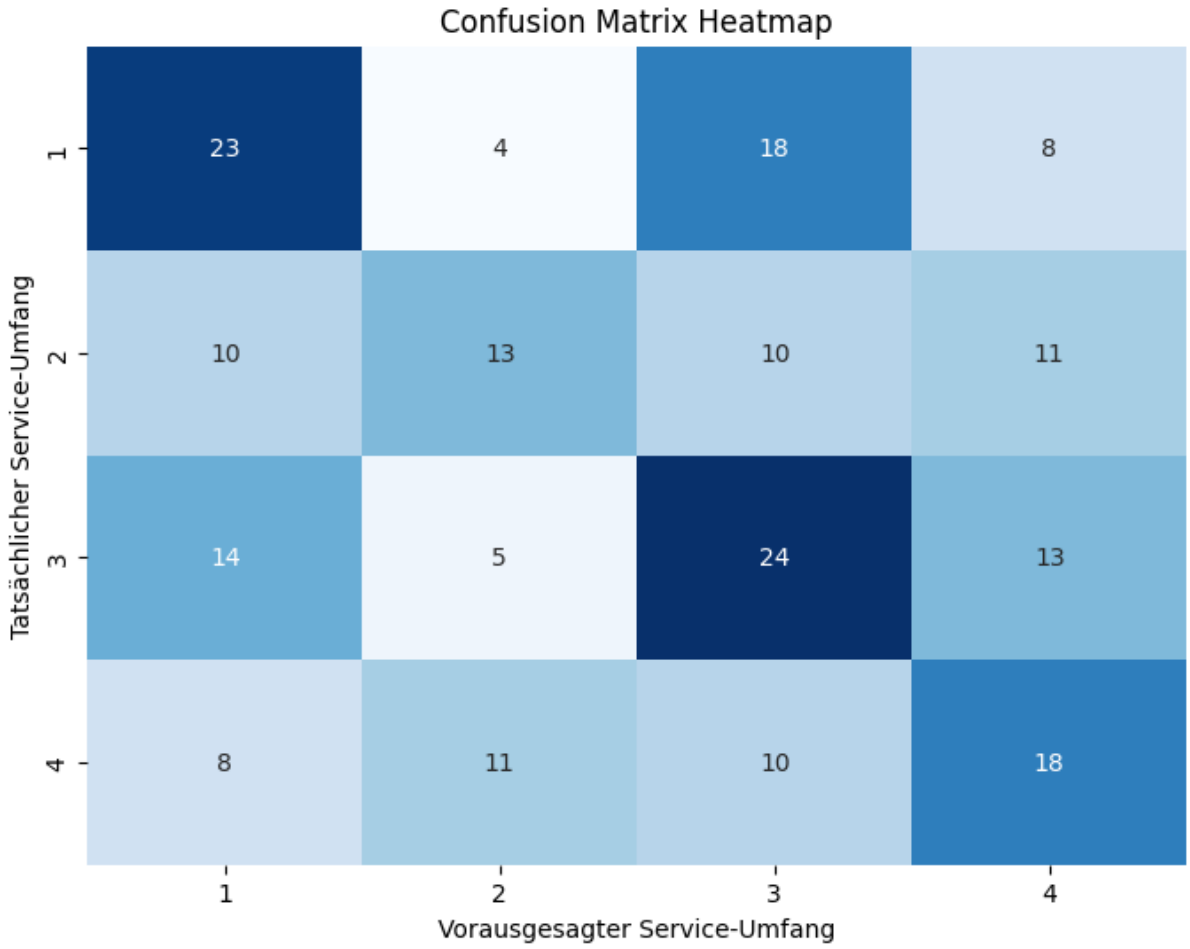

Abbildung 14: Confusion-Matrix - Random Forest

Genauigkeit: 0,3900

Klasse 1 und 3 zeigen eine vergleichsweise hohe Modellleistung. Insgesamt zeigt das Modell eine leichte Tendenz zur korrekten Klassifikation. Dennoch gibt es hohe Verwechslungen, welche insbesondere bei Klasse 2 zu erkennen sind.

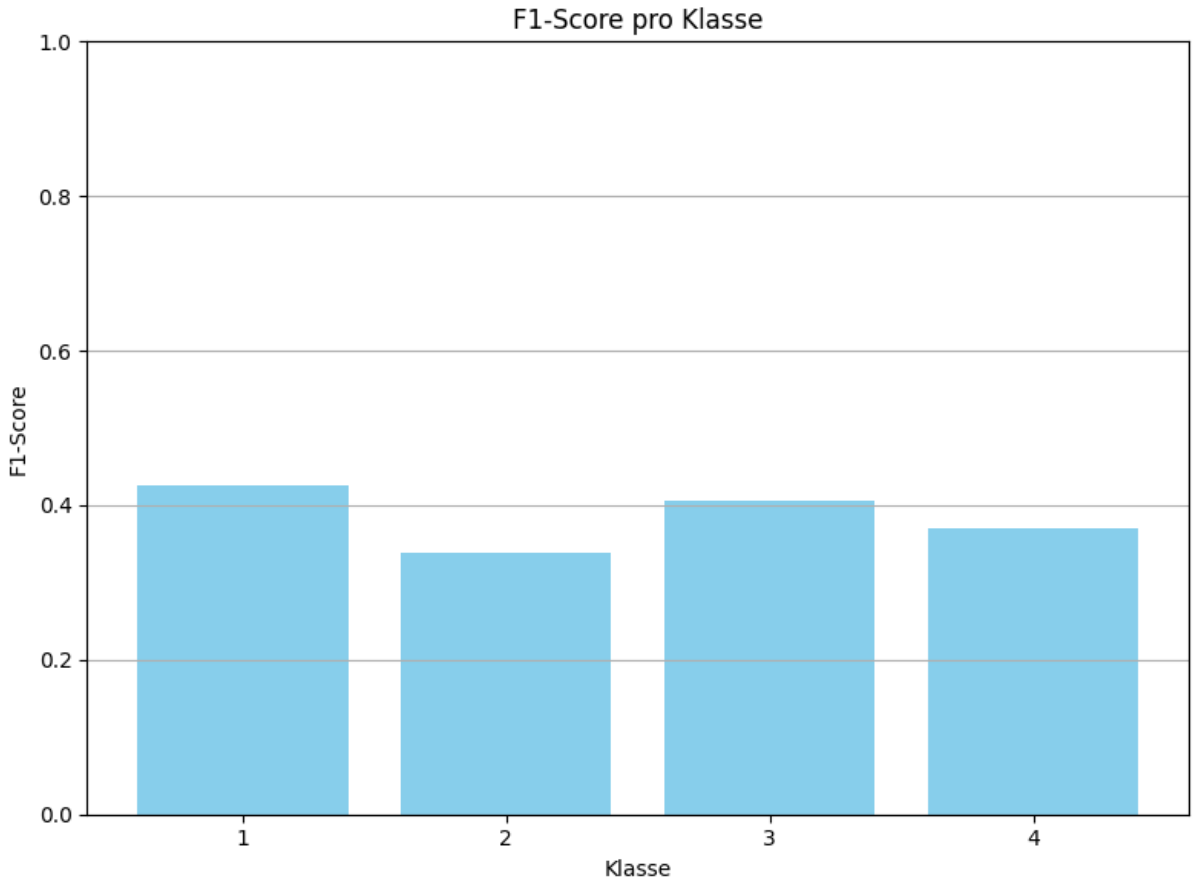

Abbildung 15: Precision pro Klasse - Random Forest

Der F1-Score kombiniert Präzision und Recall, also wie genau und vollständig ein Modell die Klassen erkennt.

Die zuvor erkennbare Schwäche von Klasse 2 wird in diesem Modell nochmals bestätigt. Klasse 1 und 3 schneiden am besten ab, was auf eine robuste Vorhersage dieser Klassen deuten lässt.

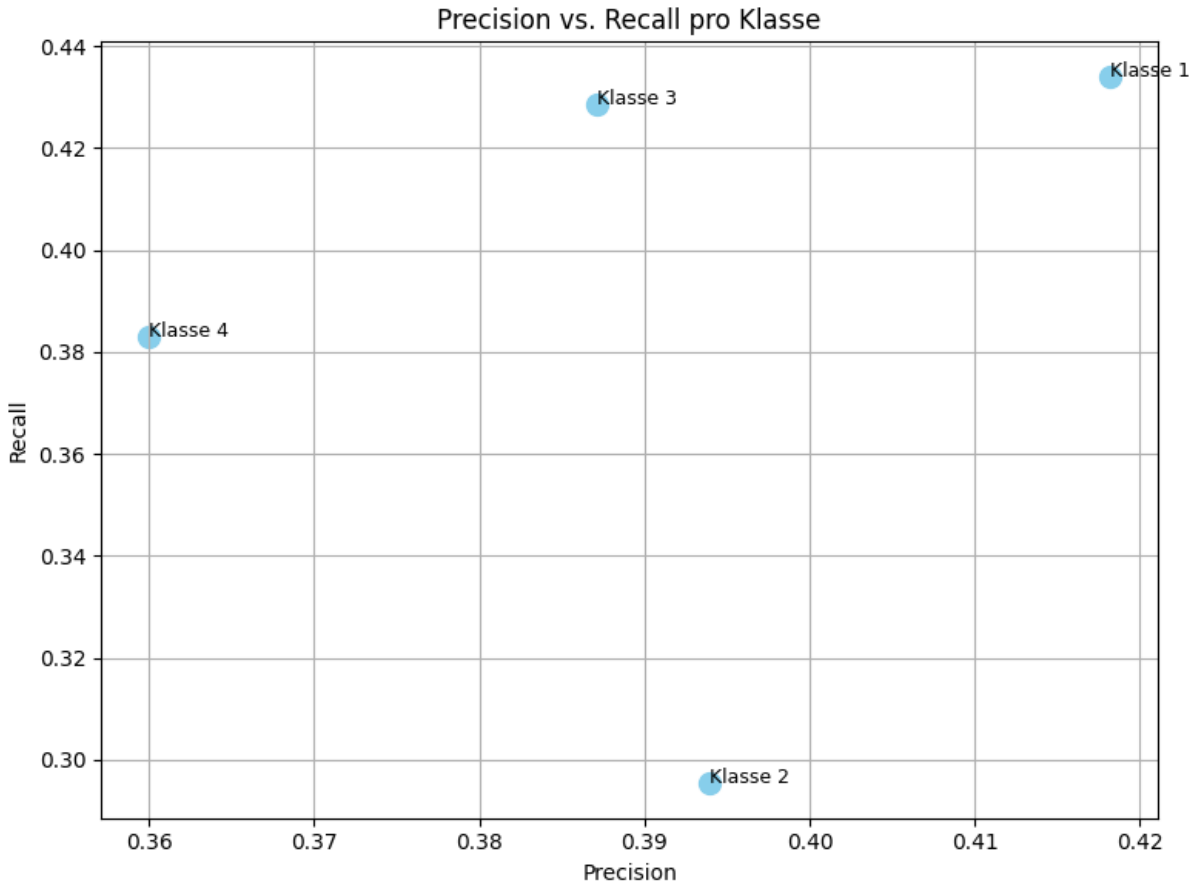

Abbildung 16: Precision vs. Recall - Random Forest

Diese Darstellung zeigt, dass Klasse 2 einen schwachen Recall aufweist. Dies bedeutet, dass viele echte Fälle nicht erkannt werden. Klasse 1 weist eine gute Balance auf und zeigt eine hohe Zuverlässigkeit und Trefferquote. Von Klasse 3 werden viele echte Fälle erkannt, aber auch mehr Fehlklassifikationen. Insgesamt wird hier ebenfalls deutlich, dass Klasse 2 abfällt. Da viele echte Instanzen der Klasse übersehen werden.

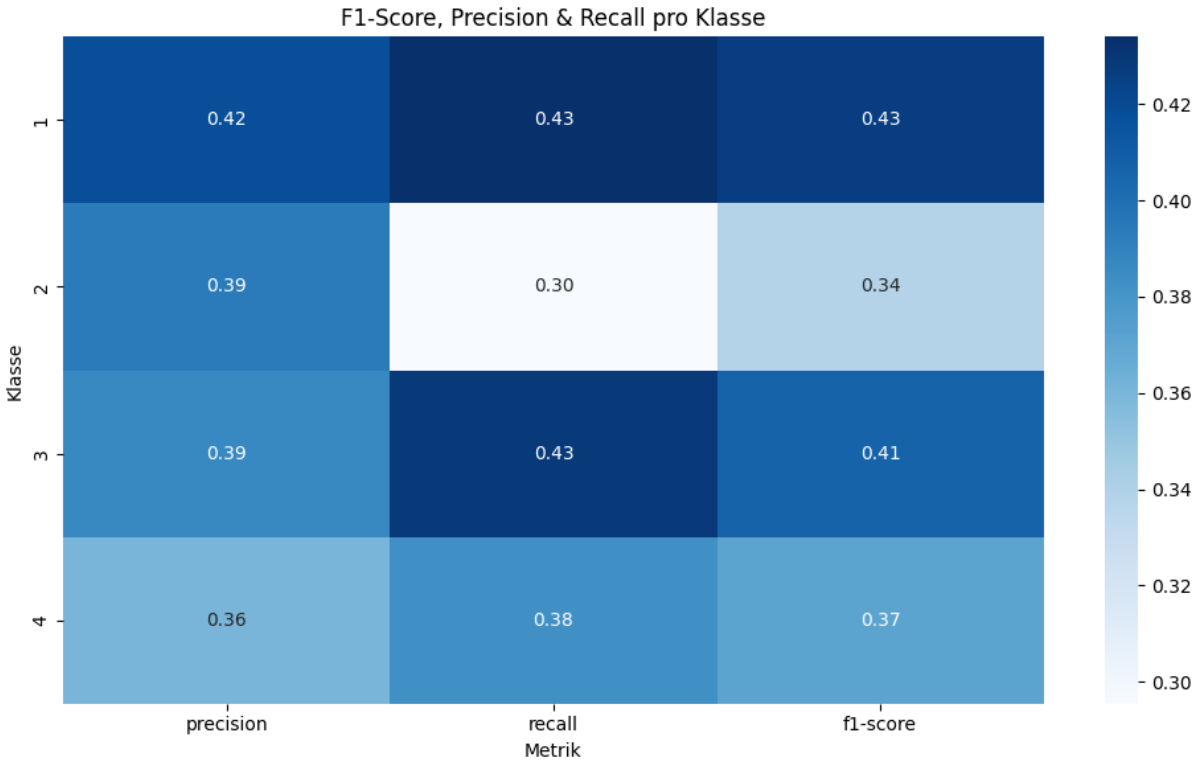

Abbildung 17: F1-Score, Precison & Recall – Random Forest

Diese Heatmap bestätigt die Muster aus dem Scatter-Plot. Klasse 1 und 3 sind stark, wobei 1 die bessere Gesamtleistung aufweist. Während Klasse 2 deutlich abfällt. Der niedrige Recall kann als Hinweis gesehen werden, dass die Klasse unterrepräsentiert ist. Demnach hat das Modell Schwierigkeiten genügend Muster zu lernen.

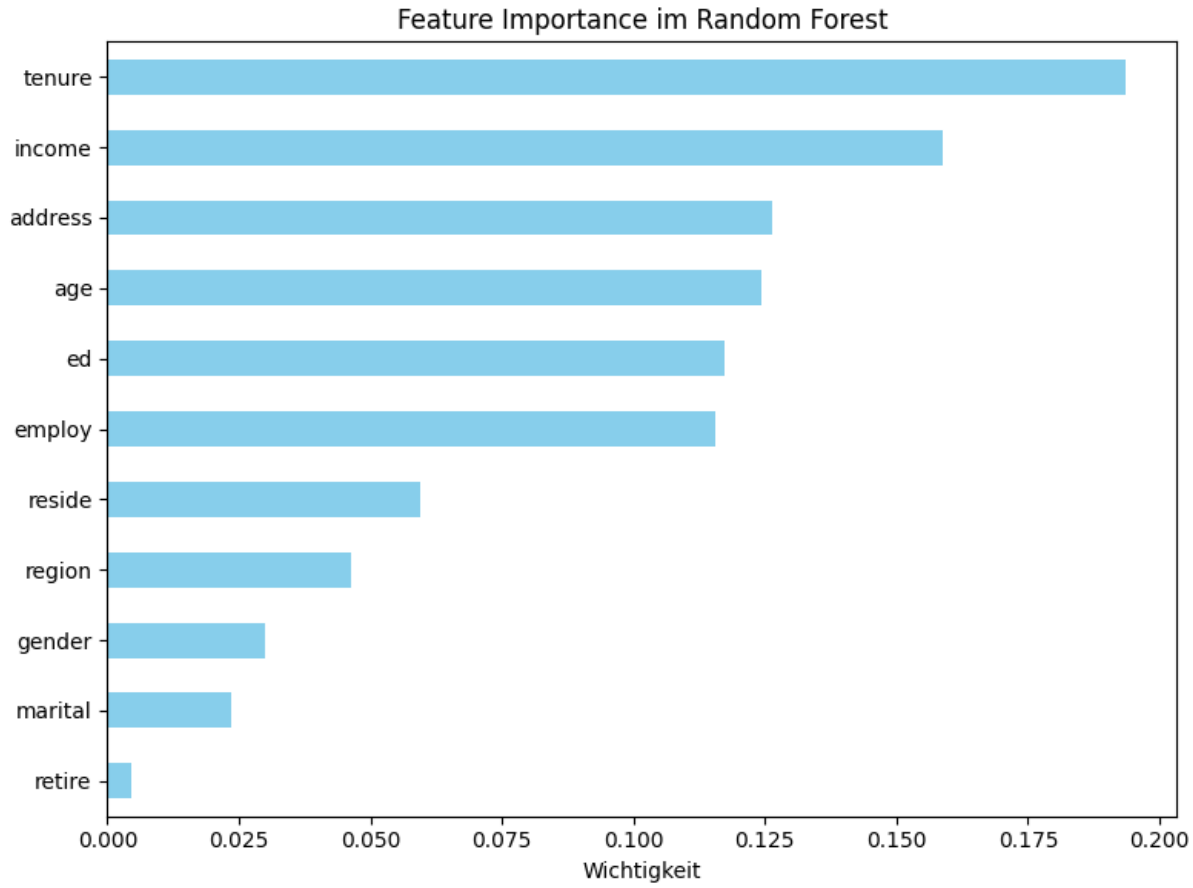

Abbildung 18: Feature Importance - Random Forest

Eine Wichtigkeit von über 0,1 erreichen die Kriterien: Tenure, Income, Adress, Age, ed und employ.

Die Reduktion des Modells auf die sechs wichtigsten Merkmale führte zu einer leichten Verbesserung der Testgenauigkeit, jedoch blieb die Kreuzvalidierungsgenauigkeit weiterhin niedrig. Dies deutet darauf hin, dass die ausgewählten Features zwar relevanter sind, aber nicht ausreichen, um die Zielvariable zuverlässig vorherzusagen. Die Ergebnisse legen nahe, dass Random Forest für diesen Klassifikationsfall nicht das optimale Modell darstellt.

4.4 Logistische Regression

Zur weiteren Analyse wird ein linearer Klassifikator verwendet, konkret die logistische Regression in ihrer multinomialen Variante. Diese Methode eignet sich für mehrklassige Klassifikationsprobleme wie die vorliegende Zielvariable custcat, die vier verschiedene Servicekategorien umfasst. Im Gegensatz zum kNN-Algorithmus, der instanzbasiert arbeitet, handelt es sich bei der logistischen Regression um ein parametrisches Verfahren, das ein statistisches Modell der Entscheidungsgrenzen auf Basis linearer Zusammenhänge zwischen den Merkmalen und der Zielvariable lernt. [8]

Vor dem Modelltraining werden die Daten mit einem StandardScaler standardisiert, da die logistische Regression empfindlich auf unterschiedlich skalierte Merkmale reagiert.[9] Anschließend wird ein Basismodell trainiert, welches in einem ersten Schritt eine mo-derate Genauigkeit auf dem Testdatensatz erzielt.

Um das Modell zu verbessern, wird ein gezieltes Hyperparameter-Tuning durchgeführt. Im Fokus steht dabei der Regularisierungsparameter C, der die Stärke der Bestrafung großer Gewichtungen kontrolliert und dadurch eine Überanpassung verhindern soll. Zusätzlich werden feste Einstellungen für Iterationsanzahl, Zufallsstartwert und Multiklassenstrategie gewählt. Die finale Parametrisierung des besten Modells ist in der nachfolgenden Tabelle dargestellt.

Tabelle 8: Parameter - Logistische Regression

Parameter	Wert
C	0.01
Max_iter	1000
Multi_class	multinominal
Random_state	1

Die Modellbewertung erfolgt wie beim kNN-Ansatz auf Basis von drei Gütekriterien: der mittleren Genauigkeit aus einer 5-fachen Kreuzvalidierung, der Genauigkeit auf

[8] (GeeksforGeeks, 2025b)
[9] (GeeksforGeeks, 2025a)

dem Trainingsdatensatz sowie der Genauigkeit auf dem abgetrennten Testdatensatz. Die Ergebnisse sind in folgender Abbildung zusammengefasst.

```
Cross-Validation Accuracy (Mittelwert): 0.4100
Genauigkeit auf Trainingsdaten: 0.4238
Genauigkeit auf Testdaten: 0.4050
```

Abbildung 19: Vergleich der Modellgenauigkeit auf Trainingsdaten, Testdaten und in der Kreuzvalidierung bei logistischer Regression

Die Abbildung zeigt nahe beieinanderliegenden Werte. Diese deuten darauf hin, dass die logistische Regression weder zu Overfitting noch zu Underfitting neigt, sondern eine stabile Generalisierungsleistung aufweist. Das Modell zeigt demnach auf unbekannten Daten eine ähnlich hohe Vorhersagequalität wie auf den Trainingsdaten, was für den Praxiseinsatz ein entscheidender Vorteil ist.

Zur tiefergehenden Bewertung des Modells wird in Abbildung 20 die Konfusionsmatrix dargestellt.

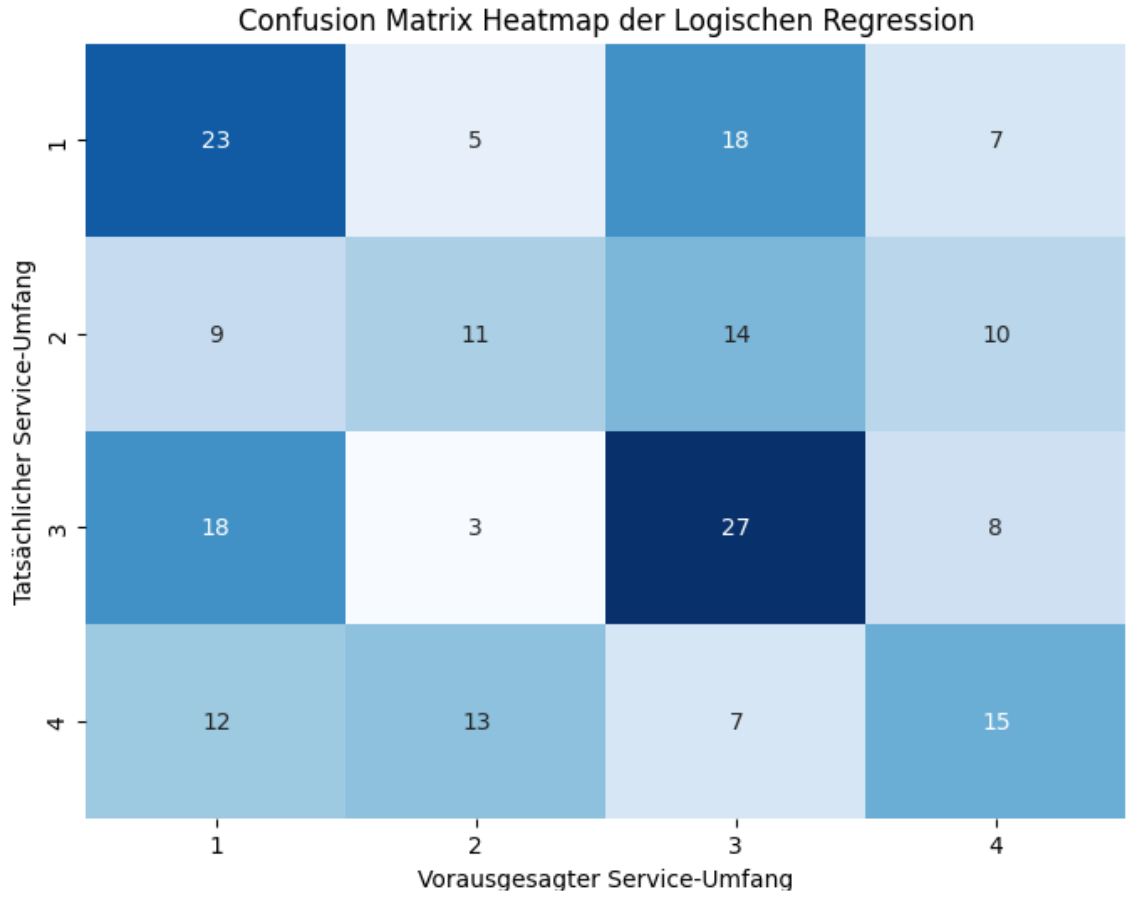

Abbildung 20: Cunfusion Matrix - Logistische Regression

Auffällig ist, dass insbesondere die Klassen 1 und 3 vergleichsweise oft korrekt erkannt werden, während es bei den Klassen 2 und 4 zu stärkeren Verwechslungen kommt.

Klasse 3 erreicht dabei mit 25 Treffern den höchsten Wert auf der Diagonalen, was auf eine solide Erkennungsrate hinweist. Gleichzeitig zeigen sich jedoch auch übergreifende Unsicherheiten, etwa bei Klasse 2, die häufig mit Klasse 3 verwechselt wird.

Zur besseren Einordnung der Klassifikationsergebnisse wird in Abbildung 21 der F1-Score pro Klasse visualisiert.

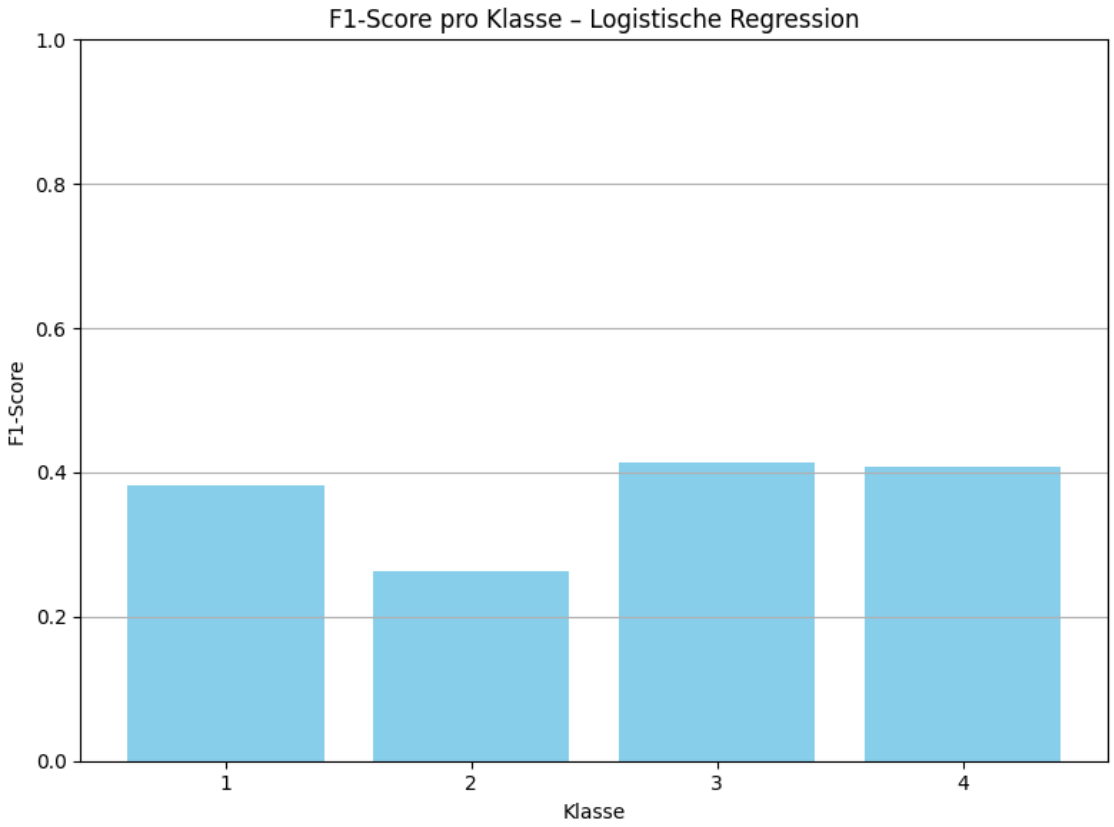

Abbildung 21: F1-Score pro Klasse - Logistische Regression

Wie erkennbar, erzielt die logistische Regression die höchsten F1-Werte für die Klassen 3 und 4, was auf eine insgesamt ausgewogene Erkennungsleistung in diesen Bereichen hinweist. Klasse 1 liegt leicht darunter und Klasse 2 erreicht den niedrigsten

Wert. Dies verdeutlicht, dass das Modell in Bezug auf Klasse 2 noch deutliche Schwächen aufweist.

Zur weiteren Analyse der Modellgüte wird der Zusammenhang zwischen Precision und Recall für jede Klasse grafisch dargestellt.

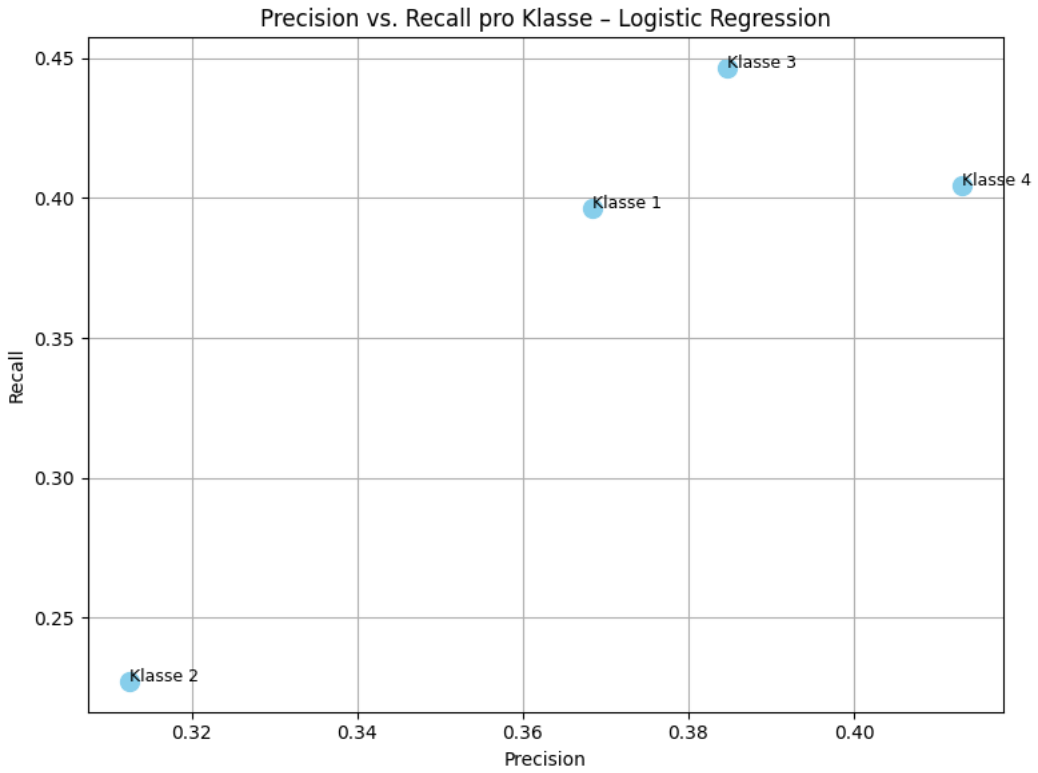

Abbildung 22: Precision vs. Recall pro Klasse – Logistische Regression

Auffällig ist Klasse 3, die sowohl bei Precision als auch Recall mit Werten über 0.45 die besten Resultate erzielt und somit insgesamt am zuverlässigsten erkannt wird. Klasse 1 liegt bei einem Recall von etwa 0.43 ebenfalls im oberen Bereich, während die Precision etwas niedriger ausfällt. Klasse 4 zeigt hingegen ein ausgewogenes,

aber mittleres Niveau in beiden Metriken. Klasse 2 weist mit einem Recall von lediglich 0.25 und einer entsprechend niedrigen Precision die schwächste Leistung auf.

Abbildung 23 bietet eine kompakte Übersicht über die drei wichtigsten Bewertungskriterien, Precision, Recall und F1-Score, getrennt nach den vier Klassen des Serviceumfangs.

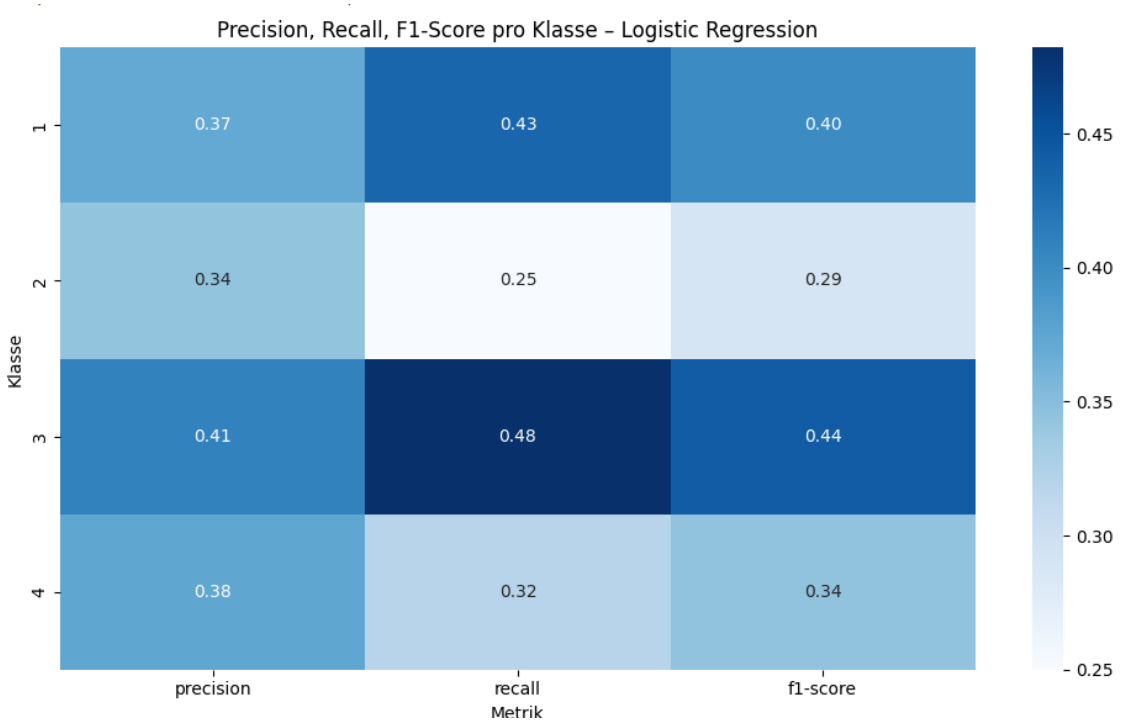

Abbildung 23: Precision, Recall und F1-Score pro Klasse – Logistische Regression

Am besten schneidet erneut Klasse 3 ab, die in allen drei Metriken mit Werten zwischen 0.41 und 0.48 heraussticht. Klasse 1 erzielt ebenfalls solide Werte, vor allem beim Recall. Klasse 2 hingegen bleibt mit einem F1-Score von lediglich 0.29 deutlich hinter den anderen zurück, insbesondere durch den geringen Recall-Wert von 0.25. Klasse 4 erreicht durchgängig mittlere Werte und liegt damit zwischen den beiden Extremen.

Zum Abschluss der Modellbewertung wird die Bedeutung einzelner Merkmale für die logistische Regression analysiert. Grundlage dafür ist der Betrag der

Modellkoeffizienten, wobei ein höherer Betrag auf einen stärkeren Einfluss auf die Klassifikation hinweist.

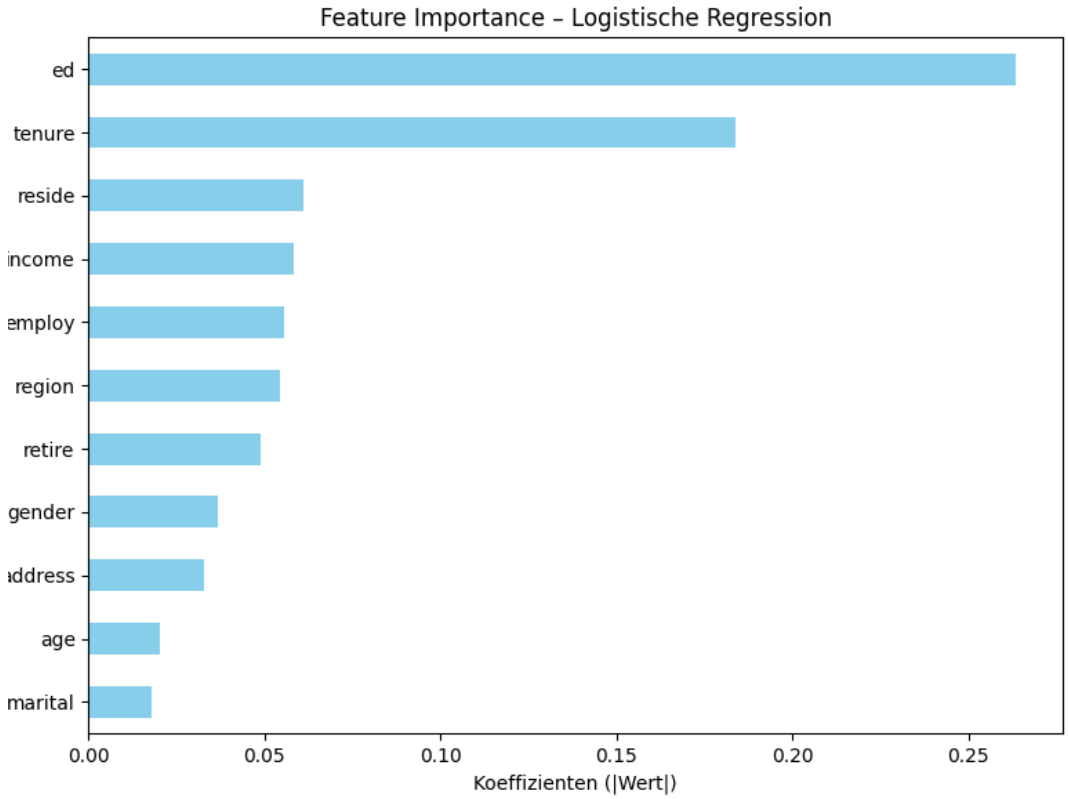

Abbildung 24: Einfluss der Merkmale auf das logistische Regressionsmodell

Deutlich wird, dass die Merkmale „ed“ (Bildungsniveau) und „tenure“ (Kundendauer) mit Abstand die größte Aussagekraft besitzen. Sie stechen mit den höchsten Koeffizientenwerten hervor und haben somit den stärksten Einfluss auf die Entscheidungsfindung des Modells. Dahinter folgen mit spürbarem Abstand Merkmale wie „reside“, „income“ oder „employ“, die eine mittlere Bedeutung aufweisen. Merkmale wie „marital“ oder „age“ liegen hingegen im unteren Bereich und tragen nur gering zur Klassifikation bei. Diese Ergebnisse bestätigen die Erkenntnisse aus dem kNN-Modell.

4.5 Support Vector Machines

Für die Implementierung des SVM-Klassifikators werden die Daten zunächst in Trainings- und Testmengen aufgeteilt und wie bei der logistischen Regression mithilfe des StandardScaler standardisiert. Dadurch erhalten alle Eingabefeatures Mittelwert 0 und Standardabweichung 1, was sicherstellt, dass keine Verzerrungen durch unterschiedlich skalierte Variablen entstehen.

Die folgende Abbildung stellt die Ergebnisse der Modellbewertung dar, basierend auf den Kriterien, welche in oberen Kapiteln bereits ausgeführt sind.

```
Cross-Validation Accuracy (Mittelwert): 0.4037
Genauigkeit auf Trainingsdaten: 0.4263
Genauigkeit auf Testdaten: 0.3950
```

Abbildung 25: Ergebnisse der Kreuzvalidierung und Vergleich der Genauigkeit der SVM auf Trainings- und Testdaten

Das erste Modell wird mit einem linearen Kernel trainiert. Es erreicht eine Trainingsgenauigkeit von 42,63 %, eine Testgenauigkeit von 39,50 % sowie eine mittlere Cross-Validation-Genauigkeit von 40,37 %. Da die Werte eng beieinanderliegen, deutet dies weder auf ein starkes Overfitting noch auf ein deutliches Underfitting hin.

Allerdings zeigt sich, dass die Gesamtgenauigkeit relativ niedrig ist, was darauf hindeutet, dass die Daten nicht linear trennbar sind und ein linearer Kernel die zugrunde liegenden Strukturen im Datensatz nicht ausreichend erfassen kann. Dieses Basismodell dient somit als Ausgangspunkt, um die Leistungsfähigkeit einer linearen Entscheidungsgrenze einzuschätzen und bildet die Grundlage für die Erprobung komplexerer Modellvarianten.

Um die Modellgüte zu verbessern, wird anschließend ein SVM-Modell mit einem RBF-Kernel eingesetzt. Die Hyperparameteroptimierung erfolgt mithilfe von GridSearchCV. Hierfür wird ein Parameterraum mit verschiedenen Werten für den Regularisierungsparameter *C* sowie den Kernel-Parameter *gamma* definiert. Beide Parameter sind entscheidend für die Modellkomplexität und die Trennschärfe der Entscheidungsgrenze. Durch eine 5-fache Cross-Validation werden alle möglichen Kombinationen dieser Parameter systematisch überprüft. Auf diese Weise liefert GridSearchCV diejenige Konfiguration, die die höchste Genauigkeit auf den Trainingsdaten erzielt und gleichzeitig eine stabile Generalisierbarkeit gewährleistet.

Tabelle 9: Optimale Parameterkombination nach GridSearchCV

Parameter	Wert
C	10
gamma	0,001

Die folgende Abbildung zeigt die erzielte Genauigkeit nach dieser Optimierung:

```
Ergebnisse des optimierten Modells:
Genauigkeit auf Trainingsdaten: 0.4250
Genauigkeit auf Testdaten: 0.4000
Beste cross-validation accuracy: 0.4150
```

Abbildung 26: Vergleich der Modellgenauigkeit des optimierten Modells auf Trainingsdaten, Testdaten und in der Kreuzvalidierung bei SVM

Die Optimierung ergibt, dass die beste Konfiguration mit C = 10 und γ = 0.001 eine Kreuzvalidierungsgenauigkeit von 41,5 % erzielt. Das daraus resultierende Modell erreicht auf den Trainingsdaten eine Genauigkeit von 42,5 % und auf den Testdaten einen Wert von 40,0 %. Damit kann im Vergleich zum linearen Basismodell zwar eine geringe Verbesserung erzielt werden, die Gesamtleistung blieb jedoch weiterhin auf einem niedrigen Niveau.

Die nahezu identischen Werte für Trainings- und Testgenauigkeit verdeutlichen, dass weder eine starke Überanpassung (Overfitting) noch eine deutliche Unteranpassung (Underfitting) vorliegt. Stattdessen zeigt das Ergebnis darauf hin, dass das Modell trotz Einsatz eines nichtlinearen RBF-Kernels und gezielter Hyperparameteroptimierung nicht in der Lage ist, die zugrunde liegenden Strukturen der Daten angemessen zu erfassen. Dies lässt sich durch stark überlappende Klassen im Merkmalsraum erklären, wodurch die Trennschärfe insgesamt begrenzt bleibt.

Folglich erweist sich das Modell in seiner aktuellen Form als nicht geeignet, um zuverlässige Vorhersagen für diesen Datensatz zu liefern.

Um die Leistung des optimierten SVM-Modells detaillierter zu bewerten, werden im Folgenden verschiedene Evaluationsmetriken und Visualisierungen dargestellt. Zusätzlich wird mit Hilfe der Feature Importance untersucht, welche Merkmale den größten Einfluss auf die Modellvorhersagen haben.

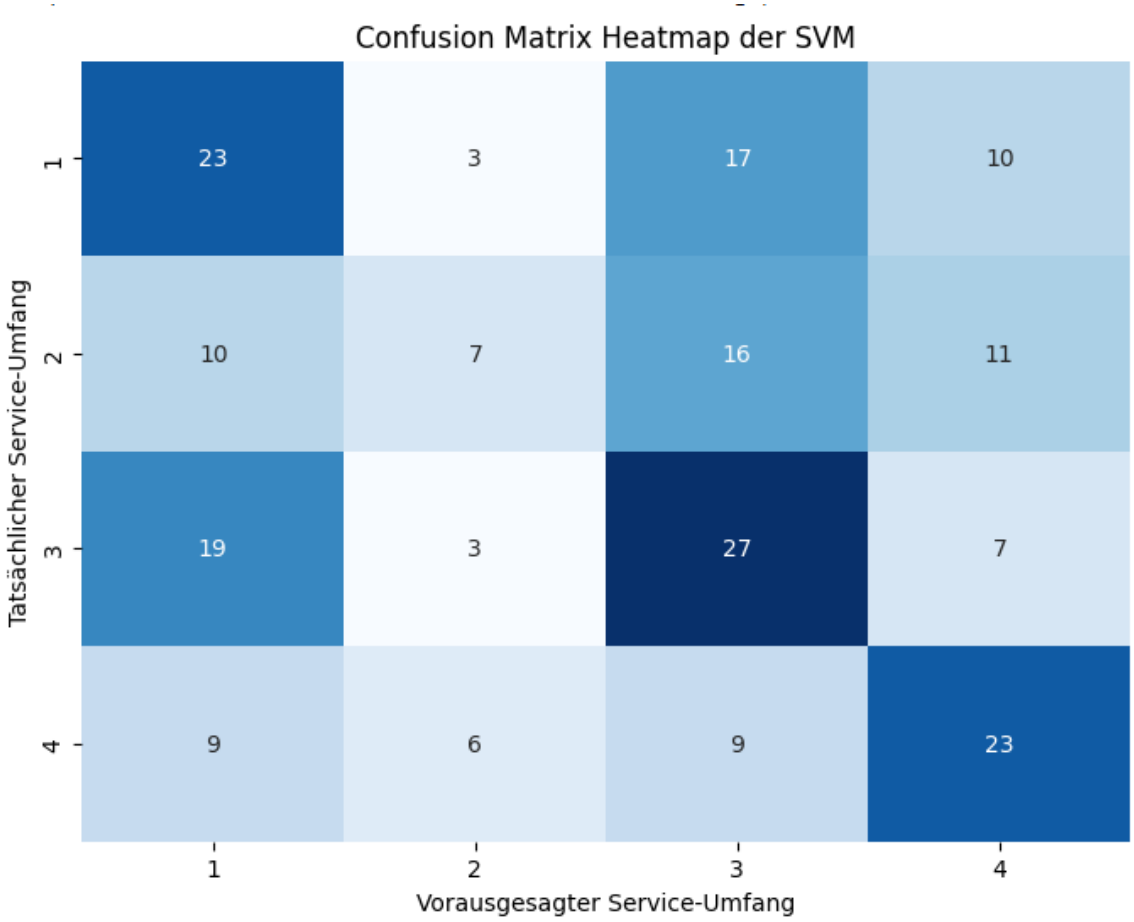

Abbildung 27: Confusion-Matrix - SVM

Das Modell erreicht insgesamt eine mittlere Trennschärfe. Am zuverlässigsten werden die Klassen 3 und 4 erkannt, was sich an den höheren Werten auf der Diagonale ablesen lässt. Besonders schwierig gestaltet sich hingegen die Unterscheidung der Klasse 2, die nahezu ebenso häufig falsch wie richtig klassifiziert wird. Darüber hinaus zeigen sich deutliche Überlappungen zwischen den Klassen 1, 3 und 4, was darauf hinweist, dass die zugrunde liegenden Merkmalsverteilungen in diesen Kategorien sehr ähnlich ausfallen.

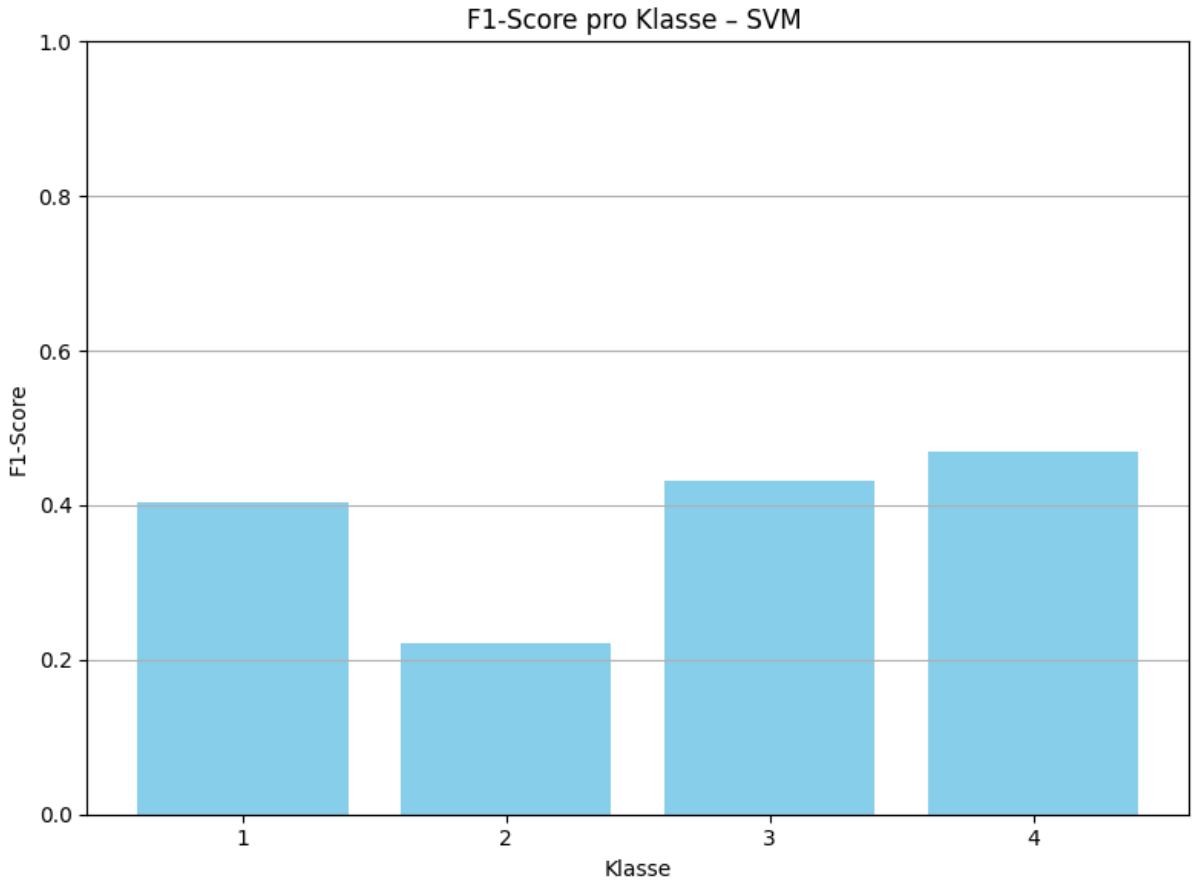

Abbildung 28: F1-Score pro Klasse - SVM

Die Abbildung zeigt die F1-Scores pro Klasse für das SVM-Modell. Dabei fällt auf, dass die Klassen unterschiedlich gut vom Modell erkannt werden.

Die höchsten Werte erreichen Klasse 4 und Klasse 3. Diese beiden Kategorien weisen somit die beste Balance zwischen Präzision und Recall auf. Klasse 1 liegt mit einem F1-Score von etwa 0.40 im Mittelfeld. Deutlich schwächer ist die Klasse 2, die mit einem F1-Score von lediglich 0.22 die geringste Modellleistung aufweist.

Damit macht die Abbildung deutlich, dass das Modell die Klassen nicht gleich gut unterscheiden kann. Während insbesondere Klasse 2 häufig fehlklassifiziert wird, gelingt die Erkennung von Klasse 3 und 4 vergleichsweise zuverlässig.

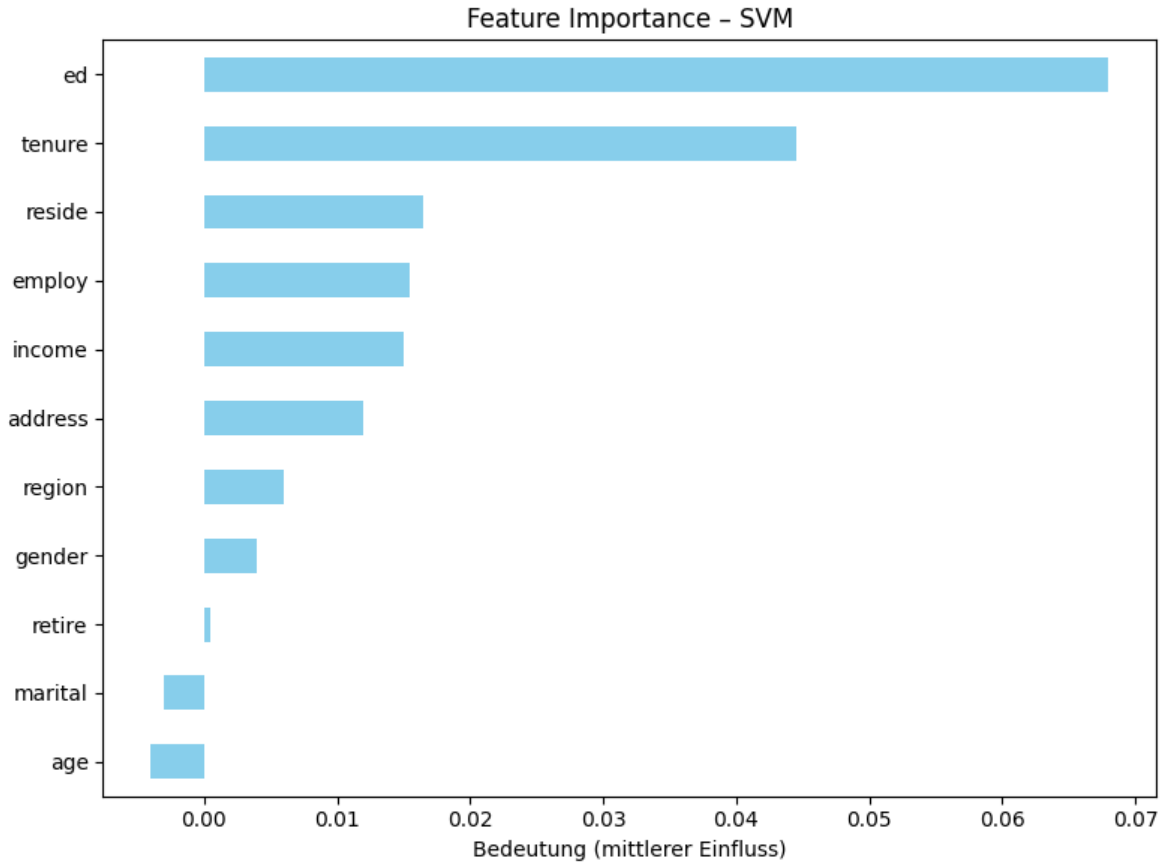

Abbildung 29: Feature Importance - SVM

Die Analyse der Feature Importance des besten SVM-Modells mit RBF-Kernel wurde mithilfe der Permutation Importance durchgeführt. Dabei zeigte sich, dass insbesondere die Variablen „ed“ (Bildung) und „tenure“ (Kundendauer) den größten Einfluss auf die Klassifikation besitzen. Merkmale wie „reside“, „employ“, „income“ und „address“ tragen in mittlerem Maße zur Modellleistung bei. Demgegenüber weisen „region“, „gender“, „marital“, „age“ sowie insbesondere „retire“ nur einen geringen Einfluss auf, sodass sie für die Vorhersage kaum relevant sind.

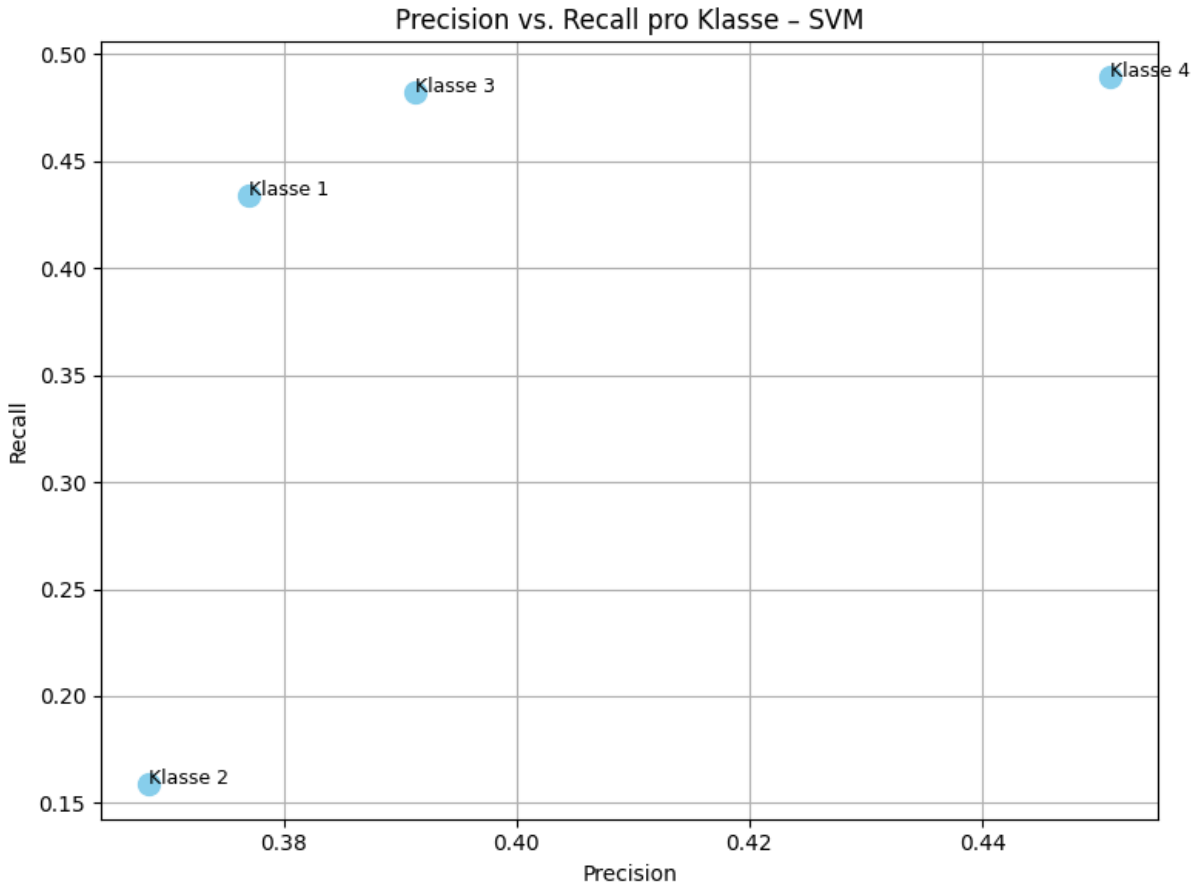

Abbildung 30: Precision vs. Recall pro Klasse – SVM

Die Abbildung stellt die Precision-Recall-Werte pro Klasse des SVM-Modells dar. Dabei zeigt sich, dass Klasse 4 sowohl bei Precision als auch bei Recall die besten Ergebnisse erzielt und somit am zuverlässigsten erkannt wird. Klasse 3 erreicht ebenfalls relativ hohe Werte und liegt knapp dahinter. Klasse 1 befindet sich im Mittelfeld mit soliden, aber schwächeren Ergebnissen. Besonders auffällig ist Klasse 2, die mit sehr niedrigen Werten in beiden Metriken die geringste Modellleistung aufweist und somit am häufigsten fehlklassifiziert wird.

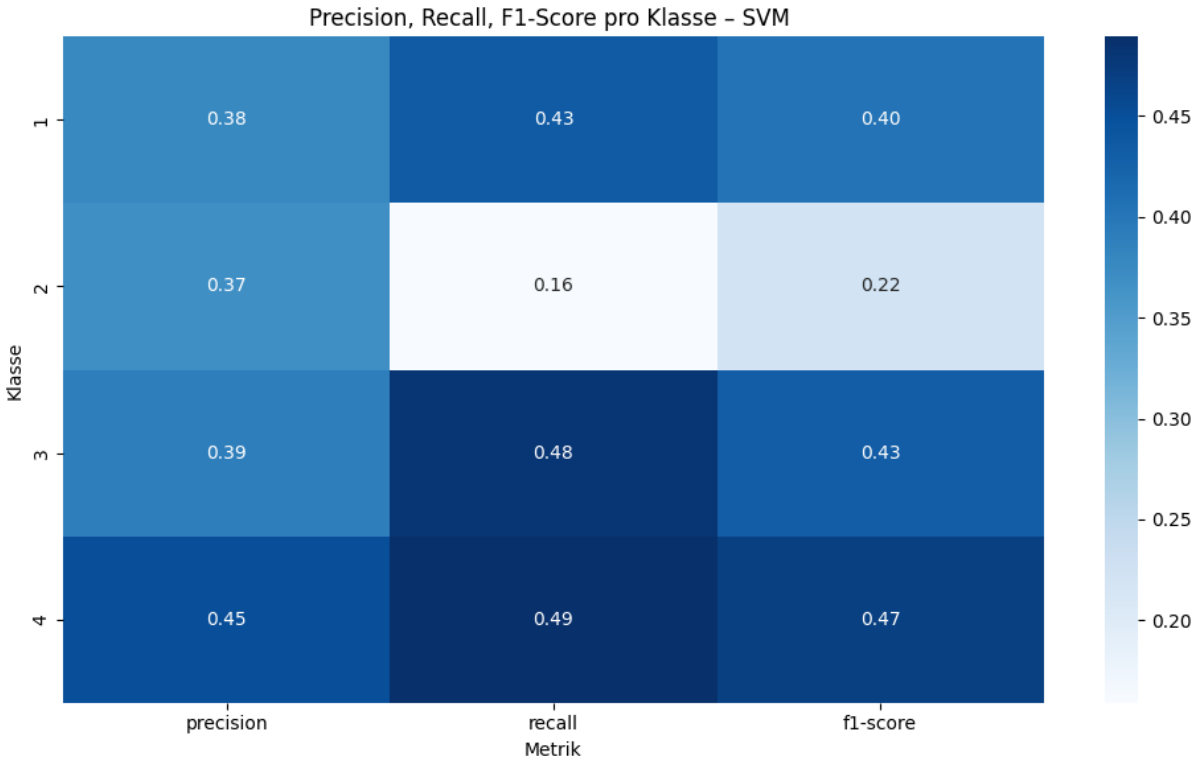

Abbildung 31: Precision, Recall und F1-Score pro Klasse – SVM

Die Heatmap zeigt die Ergebnisse für Precision, Recall und F1-Score des SVM-Modells getrennt nach Klassen. Besonders deutlich wird, dass Klasse 4 mit Werten um 0.45 bis 0.49 die besten Ergebnisse erzielt und damit am zuverlässigsten erkannt wird. Klasse 3 weist ebenfalls solide Werte auf, insbesondere beim Recall. Klasse 1 liegt im mittleren Bereich, während Klasse 2 mit sehr niedrigen Werten, insbesondere einem Recall von 0.16, die schwächste Modellleistung zeigt und am häufigsten fehlklassifiziert wird.

5. Bewertung Algorithmen

Die Bewertung der Algorithmen erfolgt anhand des F1-Scores und der Genauigkeit der Test- und Trainingsdaten.

Tabelle 10: F1-Scores der verschiedenen Klassen

Modell	Klasse 1	Klasse 2	Klasse 3	Klasse 4
kNN	0,42	0,4	0,43	0,32
Entscheidungsbaum	0,46	0,32	0,39	0,36
Random Forest	0,43	0,34	0,41	0,37
Logistische Regression	0,40	0,29	0,44	0,34
SVM	0,40	0,22	0,43	0,47

Wenn man die Modelle anhand des F1-Scores vergleicht, zeigt sich, dass kNN die insgesamt schwächste Leistung erbringt. Die F1-Werte bewegen sich nur zwischen 0,32 und 0,43, wobei besonders Klasse 4 mit 0,32 deutlich schwach erkannt wird. Der Entscheidungsbaum liefert mit F1-Scores zwischen 0,32 und 0,46 eine bessere, aber immer noch ungleichmäßige Performance. Der Random Forest erscheint auf den ersten Blick als stabiler Allrounder, da alle Klassen relativ ausgeglichen im Bereich von 0,34 bis 0,43 erkannt werden. Allerdings deutet diese gleichmäßige, aber nicht überragende Performance auf ein Overfitting hin. Das Modell passt sich stark an die Trainingsdaten an, kann diese Stabilität jedoch nicht in gleichem Maße auf die Testdaten übertragen. Daher ist der Random Forest für diesen Datensatz letztlich nicht optimal.

Die logistische Regression erzielt mit bis zu 0,44 die besten Ergebnisse, insbesondere für Klasse 3, schwächelt jedoch bei Klasse 2 (0,29). Die SVM liefert ähnlich hohe Werte wie die logistische Regression und erreicht in den Klassen 3 und 4 sogar F1-Scores bis 0,47, bricht aber bei Klasse 2 stark ein (0,22). Insgesamt ist die logistische Regression im Hinblick auf den F1-Score am ausgewogensten, während die SVM punktuell sehr stark, aber unausgeglichen ist.

Betrachtet man die Genauigkeit auf den Testdaten, erzielt die logistische Regression mit 0,4050 das beste Ergebnis. kNN und SVM liegen mit jeweils 0,4000 knapp dahinter. Der Random Forest erreicht 0,3900, während der Entscheidungsbaum mit 0,3850 das schwächste Resultat liefert.

Insgesamt zeigt sich, dass die logistische Regression sowohl beim F1-Score als auch bei der Genauigkeit der Test und Trainingsdaten, sowie auch bei der “cross-validation accuracy” die überzeugendsten Ergebnisse liefert. Die SVM ist in einzelnen Klassen ebenfalls stark, weist jedoch deutliche Schwächen bei Klasse 2 auf. Der Random Forest wirkt zwar stabil, leidet aber unter Overfitting. Der Entscheidungsbaum schneidet etwas besser ab als kNN, bleibt jedoch insgesamt hinter den stärkeren Verfahren zurück. kNN zeigt in beiden Vergleichen die schwächste Leistung und eignet sich für diesen Datensatz nicht.

6. Beantwortung Fragestellung

Um die große Fragestellung nun final zu beantworten, wird untersucht, welche Kundenmerkmale den größten Einfluss auf den gewünschten Serviceumfang haben und wie diese Unterschiede sichtbar gemacht werden können. Dazu werden die in Kapitel 3 bereits erwähnten Fragestellungen nun im Folgenden nochmals aufgegriffen.

Welche Merkmale sind typisch für Kunden mit hohem Serviceumfang und welche neuen Kunden ähneln, bestehenden hohen Service Kunden? Die logistische Regression hat sich in dieser Analyse als leistungsstärkstes Modell gezeigt. Sie verdeutlicht, dass Kunden mit hohem Serviceumfang (besonders aus Klasse 3) ein höheres Bildungsniveau sowie eine längere Kundendauer aufweisen. Diese beiden Merkmale besitzen die höchsten Modellkoeffizienten und sind somit entscheidend für die Klassifikation.

Die folgende Abbildung veranschaulicht die Top-5 Merkmale für Klasse 4 anhand ihrer Modellkoeffizienten:

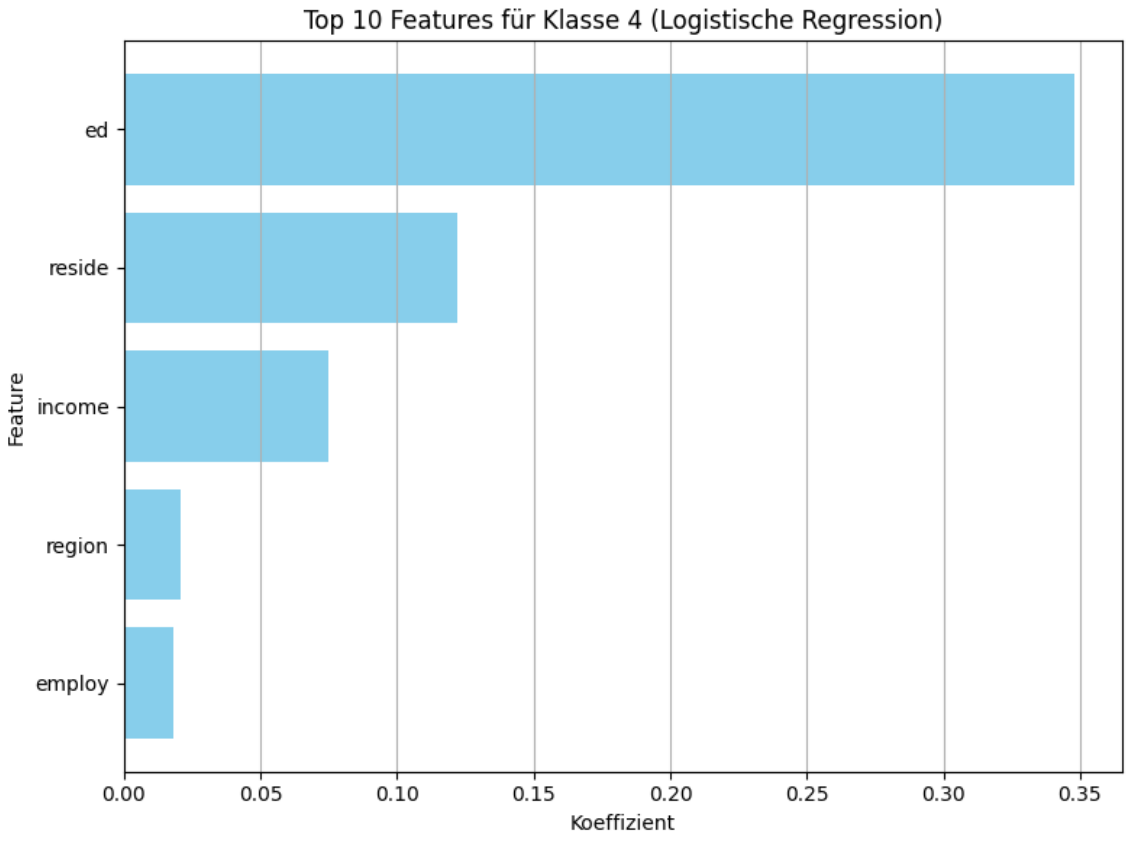

Abbildung 32: Top-10 Features für Klasse 4 - Logistische Regression

Neue Kunden, die in diesen zentralen Merkmalen ähnliche Werte zeigen, lassen sich mit hoher Wahrscheinlichkeit ebenfalls der Klasse mit hohem Serviceumfang

zuordnen. Die Erkenntnisse können vor allem für einen gezielten Vertrieb und eine kundenorientierte Servicegestaltung eine wichtige Grundlange darstellen. Verbindet man diese Darstellung nun mit der Analyse des Datensatzes, lassen sich nachfolgende Ergebnisse ableiten. Neue Kunden, welche ein hohes Bildungsniveau, schon länger in ihrer Wohnregion ansässig sind und ein hohes Einkommen aufweisen., sind höchstwahrscheinlich in Klasse 4 zuzuordnen. Demnach wünschen Kunden mit diesen Merkmalen, einen hohen Service-Umfang.

Wie unterscheiden sich die Kundengruppen? Die logistische Regression liefert zum einen die höchste Gesamtgenauigkeit der getesteten Algorithmen und bietet auch eine gute Differenzierung zwischen den Serviceklassen (vor allem der Klasse 3). Die Auswertung der Merkmalsgewichte zeigt, dass sozioökonomische Faktoren wie Bildung, Einkommen und Beschäftigungsdauer die größten Unterschiede zwischen den Kundengruppen erklären. Im Gegensatz dazu haben dem Modell zufolge Region und Geschlecht nur einen geringen Einfluss auf die Serviceklassifikation. Daraus ergibt sich, dass Marketingmaßnahmen besser auf individuelle, verhaltens- und statusbezogene Merkmale ausgerichtet werden sollten, statt auf demografische Kategorien. Die logistische Regression liefert hierfür klare Indikatoren und unterstützt damit eine effizientere Zielgruppenansprache.

Gibt es Unterschiede zwischen Regionen und Geschlechter in Bezug auf Servicewünsche? Die Analyse der einzelnen Modelle und den Einfluss der unterschiedlichen Features, ergab, dass sowohl Region als auch Geschlecht kaum bis wenig Einfluss auf die einzelnen Serviceumfänge hat. Hier lassen sich demnach keine besonderen Marketingmaßnahmen für einzelne Regionen oder Geschlechter erschließen. Ein gezielter Vertrieb in Regionen mit hohem Serviceumfang, lässt sich demnach nicht umsetzten und würde nicht für mehr Neukunden mit gewünschtem Serviceumfang sorgen. Eine gezielte Kommunikations- oder Vertriebsstrategie, welche ausschließlich sich auf diese Kriterien stürzt, wäre demnach nicht zielführend. Stattdessen sollten Merkmale mit höherer Bedeutung, wie etwas Einkommen stärker in Segmentierung und Ansprachen einbezogen werden.

Kann ein neuer Kunde zuverlässig in eine Kategorie eingeordnet werden? Diese Frage lässt sich mit eher weniger beantworten. Da alle trainierten und getesteten Modelle eine Genauigkeit von 40,5% oder weniger aufweisen, was nicht als zuverlässig angesehen werden kann. Hier ist allerdings deutlich zu sagen, dass die Genauigkeit durch

mehr Daten erhöht werden könnte und eine zuverlässige Vorhersage geschaffen werden kann.

7. Fazit

Die Modelle zeigen auf dem gegebenen Telefonkunden-Datensatz nur begrenzte Vorhersagekraft. Die beste Testgenauigkeit erreicht die logistische Regression mit 0,4050. SVM und kNN folgen mit 0,4000. Random Forest liegt bei 0,3900, der Entscheidungsbaum bei 0,3850. Im F1-Vergleich ist Klasse 2 durchgängig am schwächsten, während Klasse 3 und teilweise Klasse 4 besser erkannt werden. Die wichtigsten Merkmale über alle Auswertungen hinweg sind ed und tenure, danach income, employ und reside. Die Region und das Geschlecht spielen kaum eine Rolle.

Für die Praxis eignet sich die logistische Regression als Basismodell, da sie die beste Kombination aus Stabilität und Leistung bietet. SVM ist eine brauchbare Alternative, zeigt jedoch deutliche Schwächen bei Klasse 2. Random Forest wirkt stabil, überträgt die hohe Trainingsleistung jedoch nicht auf neue Daten. Entscheidungsbaum und kNN sind insgesamt unterlegen.

Die Grenzen der Analyse liegen vor allem in der kleinen Stichprobe von 1.000 Fällen und der ungleichen Klassenverteilung. Beides reduziert die Generalisierbarkeit. Um die Prognosegüte spürbar zu erhöhen, sind folgende Schritte sinnvoll: mehr und aktuellere Daten, gezielte Balance der Klassen, Feature-Engineering mit Fokus auf die identifizierten Schlüsselfaktoren, Kosten-sensitive Lernverfahren sowie der Vergleich zusätzlicher Modelle wie Gradient Boosting. Für einen produktiven Einsatz empfiehlt sich eine probabilistische Ausgabe mit Kalibrierung, klare Schwellen pro Klasse und ein Monitoring der Modellleistung im Betrieb.

Mit den vorliegenden Daten liefert die logistische Regression die insgesamt überzeugendsten Ergebnisse. Für belastbare, operativ nutzbare Vorhersagen braucht es jedoch datenseitige Erweiterungen und ein stärker regularisiertes, auf Klasse 2 optimiertes Modell.

Literaturverzeichnis

GeeksforGeeks. (2025a, Juli 23). *Logistic Regression and the Feature Scaling Ensemble*. https://www.geeksforgeeks.org/machine-learning/logistic-regression-and-the-feature-scaling-ensemble/

GeeksforGeeks. (2025b, August 2). *Logistic Regression in Machine Learning*. https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/

Höchenberger, R. (2025, Januar 7). *K-Nearest Neighbors (kNN)* [Vorlesungsfolien].

Lewis, R. J. (2000). *An Introduction to Classification and Regression Tree (CART) Analysis*.

scikit-learn developers. (o. J.-a). *1.10. Decision Trees*. Scikit-Learn. Abgerufen 12. September 2025, von https://scikit-learn/stable/modules/tree.html

scikit-learn developers. (o. J.-b). *DecisionTreeClassifier*. Scikit-Learn. Abgerufen 12. September 2025, von https://scikit-learn/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

scikit-learn. (o. J.-c). *KNeighborsClassifier*. Scikit-Learn. Abgerufen 8. September 2025, von https://scikit-learn/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

Shams, M. Y., Elshewey, A. M., El-kenawy, E.-S. M., Ibrahim, A., Talaat, F. M., & Tarek, Z. (2024). Water quality prediction using machine learning models based on grid search method. *Multimedia Tools and Applications*, *83*(12), 35307–35334. https://doi.org/10.1007/s11042-023-16737-4

Taufiq, A., Yulianti, S., Rahmatulloh, A., Darmawan, I., & Rizal, R. (2024). Comparison of Hyperparameter Tuning Techniques on KNN Algorithm to find the Best K Value using Grid Search and Random Search Methods. *2024 7th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 180–186. https://doi.org/10.1109/ISRITI64779.2024.10963519