FernUniversität in Hagen
Fakultät für Mathematik und Informatik

Artificial Intelligence Group

Analyse von Kanalzustandsinformationen mit Machine-Learning-Verfahren und Erklärbarkeit der Ergebnisse

Masterarbeit

zur Erlangung des Grades eines Master of Science (M.Sc.)
im Studiengang Praktische Informatik

vorgelegt von
Karl Heinz Wichmann

Erstgutachter:	Prof. Dr. Matthias Thimm Artificial Intelligence Group
Betreuer:	Britta Sennewald Bundesamt für Sicherheit in der Informationstechnik

Erklärung

Ich erkläre, dass ich die Masterarbeit selbstständig und ohne unzulässige Inanspruchnahme Dritter verfasst habe. Ich habe dabei nur die angegebenen Quellen und Hilfsmittel verwendet und die aus diesen wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht. Die Versicherung selbstständiger Arbeit gilt auch für enthaltene Zeichnungen, Skizzen oder graphische Darstellungen. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder derselben noch einer anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht. Mit der Abgabe der elektronischen Fassung der endgültigen Version der Arbeit nehme ich zur Kenntnis, dass diese mit Hilfe eines Plagiatserkennungsdienstes auf enthaltene Plagiate geprüft werden kann und ausschließlich für Prüfungszwecke gespeichert wird.

Der Veröffentlichung dieser Arbeit auf der Webseite des Lehrgebiets Künstliche Intelligenz und damit dem freien Zugang zu dieser Arbeit stimme ich ausdrücklich zu.

Für diese Arbeit erstellte Software wurde quelloffen verfügbar gemacht, ein entsprechender Link zu den Quellen ist in dieser Arbeit enthalten. Gleiches gilt für angefallene Forschungsdaten.

Straubing, 01.07.2024 

(Ort, Datum) (Unterschrift)

Hinweis zum Datenschutz: Aus Datenschutzgründen erfolgte keine Weitergabe der aufgezeichneten Kanalzustandsinformationen an die Fernuniversität Hagen. Die erstellten Scripte und Ergebnisse dieser Arbeit sind unter folgendem Link verfügbar:
https://github.com/wichmannkarl/Masterthesis

Zusammenfassung

Die Sicherheit von Informationen gewinnt immer mehr an Bedeutung, da die Speicherung und Verarbeitung persönlicher Daten eine zentrale Rolle einnehmen. Die zunehmende Bereitstellung freier WLAN-Netzwerke erlaubt die einfache Verbindung mobiler Geräte mit einem WLAN-Netzwerk. Um eine stabile WLAN-Verbindung zu gewährleisten, tauschen diese Kanalzustandsinformationen aus, die sich in Abhängigkeit von der Umgebung verändern. Mithilfe von Machine-Learning-Algorithmen können diese Änderungen klassifiziert werden. Dies birgt jedoch die Gefahr eines möglichen Ausspähens von Benutzerkennung und Passwörtern. Die nachfolgende Arbeit beschäftigt sich mit dem Ziel, Kanalzustandsinformationen mit Hilfe von Maschinen-Learning-Algorithmen möglichst gut zu kassifizieren und die Ergebnisse zu erklären. Zunächst erfolgte in einem ersten Schritt die Verarbeitung der Kanalzustandinformationen, um anschließend mit diesen ausgewählte Machine-Learning-Algorithmen zu trainieren und Vorhersagen zur Klassifizierung von Bewegung zu treffen. Im Anschluss wurden Erklärbarkeitsmethoden eingesetzt, um zu bestimmen, welche Kanalzustandinformationen den größten Einfluss auf die Entscheidungen der Machine-Learning-Modelle haben. Es konnte festgestellt werden, dass der XGBoost-Algorithmus eine hohe Leistung und Performance, insbesondere durch den Einsatz der Grafikprozessoreinheit (GPU), aufweist. Die Erklärbarkeit der Ergebnisse wurde mit Explainable Boosting Machine und SHAP analysiert. Die Ergebnisse der beiden Erklärbarkeitsmethoden sind sehr ähnlich und zeigen die Merkmale den größten Einfluss auf die Vorhersagen der Machine-Learning-Modelle haben.

Abstract

The security of information is becoming increasingly important as the storage and processing of personal data takes on a central role. The increasing availability of free Wi-Fi networks makes it easy to connect mobile devices to a Wi-Fi network. To ensure a stable Wi-Fi connection, these devices exchange channel status information that changes depending on the environment. Machine learning algorithms can be used to classify these changes. However, this harbours the risk of possible spying on user IDs and passwords. The following work deals with the goal of classifying channel status information as well as possible with the help of machine learning algorithms and explaining the results. The first step was to process the channel status information in order to subsequently train selected machine learning algorithms and make predictions for classifying movement. Subsequently, explainability methods were used to determine which channel state information has the greatest influence on the decisions of the machine learning models. It was found that the XGBoost algorithm has high power and performance, especially through the use of the graphics processing unit (GPU). The explainability of the results was analysed using Explainable Boosting Machine and SHAP. The results of the two explainability methods are very similar and show which features have the greatest influence on the predictions of the machine learning models.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

In den vergangenen Jahren konnten deutliche Fortschritte in der Entwicklung der Digitalisierung beobachtet werden, welche weitreichende Auswirkungen auf nahezu alle Lebensbereiche, von Wirtschaft über Bildung bis hin zu Kommunikation und sozialen Interaktionen, hatten. Notebooks, Tablets und Smartphones werden zunehmend in unserer Gesellschaft genutzt, um auf einfache Weise auf das Internet und dessen Dienste zugreifen zu können. Die Verfügbarkeit öffentlicher WLANs oder Gäste-WLANs nimmt zu, sodass in vielen Bereichen eine WLAN-Verbindung aufgebaut werden kann, mit der sich mobile Geräte verbinden lassen. Dies betrifft auch Städte, Kommunen und Hotels, die ihren Kunden und Gästen entsprechende Zugänge zur Verfügung stellen. Die WLAN-Technik verwendet eine leistungsfähige Technologie, um eine stabile Signalqualität zwischen Sender und Empfänger zu gewährleisten [1]. Zu diesem Zweck werden Kanalzustandsinformationen (Channel-State-Information - CSI) [2] über die drahtlose Datenverbindung übertragen. Die Eigenschaften des WLAN-Signals werden durch die Umgebung beeinflusst, da Objekte und Personen das Signal verändern. Diese Eigenschaft erlaubt die Erkennung von Bewegungen mit Hilfe der Kanalzustandsinformationen [3] [4]. Diese Tatsache birgt das Risiko eines möglichen Sicherheitsproblems im Bereich der Datensicherheit, da die Informationen dazu verwendet werden könnten, um Benutzerdaten wie Benutzernamen und Passwörter aufzuzeichnen [5]. Eine weitere Gefahr besteht in der Erstellung von Bewegungsmustern und der Zuordnung von Bewegungsprofilen. Auf diese Weise könnten in Firmen Leistungsprofile von Mitarbeitern erstellt werden. Die Nutzung von Kanalzustandsinformationen eröffnet jedoch auch neue Möglichkeiten im Bereich der Sicherheit und der Bedienerfreundlichkeit. Mögliche Einsatzbereiche stellen beispielsweise die Bewegungserkennung in Gebäuden als Ersatz für eine Einbruchsmeldeanlage ohne optische oder thermische Sensoren, sowie die Detektion von Stürzen hilfebedürftiger Personen dar [6]. Mit Hilfe von Machine-Learning-Algorithmen soll eine Klassifizierung der Kanalzustandsinformationen erfolgen, um Bewegungsabläufe zu unterscheiden. Darüber hinaus soll mit Erklärbarkeitsmethoden analysiert werden, welche Kanalzustandsinformationen maßgeblich für die Ergebnisse der Klassifizierung ausschlaggebenden sind.

1.2 Ziel der Arbeit

Die vorliegende Masterarbeit verfolgt zwei Ziele. Einerseits soll die Klassifikation von Kanalzustandsinformationen mithilfe von Machine-Learning-Algorithmen erfolgen, um die Eignung der Algorithmen für unterschiedliche Anwendungsfälle zu analysieren. Andererseits sollen Erklärbarkeits-Algorithmen eingesetzt werden, um die resultierenden Ergebnisse zu erklären und den Einfluss von Trägerinformationen auf die Klassifizierungsergebnisse zu bestimmen. Die Basis für die Analyse der Daten stellen die vom BSI zur Verfügung gestellten Kanalzustandinformationen dar. Diese wurden in einer Versuchsumgebung des BSI mit Hilfe von Mikrocontrollern ermittelt und aufgezeichnet. Zur Vorverarbeitung, Datenanalyse und Durchführung des maschinellen Lernens wird, soweit möglich, die Open-Source-Bibliothek scikit-learn verwendet. Die Ergebnisse der Klassifizierung sollen dazu beitragen, eine Übersicht über die geeignetsten Algorithmen zu ermitteln, mit denen die Unterscheidung von Bewegungen möglich ist. Zudem soll mit unterschiedlichen Methoden der Erklärbarkeit ein Modell mit einer möglichst hohen Vorhersagequalität bestimmt werden. Die Erklärbarkeit der Ergebnisse erfolgt auf Basis eines White- und Backbox-Modells. Dadurch soll eine bessere Bewertbarkeit der Ergebnisse gewährleistet werden. Das Ziel ist es letztlich, die Entscheidung eines Modells nachhaltig zu verstehen.

1.3 Aufbau der Arbeit

Zunächst werden die Grundlagen der digitalen Modulationsverfahren und der Antennentechnik erläutert. Im Anschluss daran erfolgt die Erläuterung der Kanalzustandsinformationen, die für das Verständnis dieser Arbeit von Nutzen sind. Des Weiteren erfolgt eine Beschreibung der Methoden des maschinellen Lernens, sowie eine Auswahl und Darstellung der Machine-Learning-Algorithmen, die zum Einsatz kommen. Im Anschluss werden Erklärungsansätze für das maschinelle Lernen und deren Methoden und Prinzipien erörtert. Im Folgenden wird der Aufbau der Versuchsumgebung, sowie die Beschreibung der Bewegungsabläufe dargelegt. Nach der Vorverarbeitung der Datensätze erfolgt die Anwendung der verschiedenen Machine-Learning-Algorithmen auf die Datensets. Darauf aufbauend erfolgt die Bewertung der Ergebnisse, sowie die Zuordnung der Algorithmen und Bewegungen zu Use-Cases. Im Anschluss erfolgt die Anwendung von Methoden der Erklärbarkeit auf die Datensets, um die Zusammenhänge und Muster transparent darzustellen. Dadurch kann die Einflussnahme der Trägerinformationen auf die Ergebnisse ermittelt werden. Der letzte Abschnitt zieht das Fazit der ermittelten Erkenntnisse.

2 Theoretische Grundlagen

Im folgenden Kapitel werden die theoretischen Grundlagen der drahtlosen Datenübertragung erläutert. Der Schwerpunkt liegt auf dem Institute of Electrical and Electronics Engineers (IEEE) 802.11 Standard, der die Grundlage für Wireless Local Area Network (WLAN) bildet. Zur weiteren Untersuchung der Kanalzustandsinformationen (Channel State Information - CSI) dienen diese als technische Grundlage. Der erste Abschnitt dieses Kapitels gibt eine Einführung in die digitalen Modulationsarten des 802.11g-Standards und erläutert das MIMO-Übertragungssystem für die drahtlose Kommunikation. Der darauffolgende Abschnitt beschäftigt sich mit dem Aufbau des WLAN-Signals, sowie den Kanalzustandsinformationen.

2.1 Digitale Modulationsverfahren

Bei einer WLAN-Verbindung unterscheidet man zwischen der Modulation des Hauptträgers nach dem IEEE-Standard 802.11 und der Modulation von Unterträgern nach 802.11(a/g/n/ac) [7]. Dieser Standard sieht für die Übertragung digitaler Informationen die Phasen- oder Amplitudenmodulation vor [8]. Die Übertragung von digitalen Daten erfolgt mit Hilfe von Phase-Shift Keying (PSK), Quadrature Phase-Shift Keying (QPSK) und Quadratur Amplituden Modulation (QAM). [9]

2.1.1 Phase-Shift Keying

Die PSK-Modulation ist ein Phasenmodulationsverfahren bei dem die Phasenlage des Trägersignals die Information zum Zeitpunkt t bei gleicher Amplitude bestimmt [10]. Die Phasenlage stellt somit die zu übertragende Information dar. Das phasenumgetastete Signal kann wie folgt beschrieben werden:

$$x_c(t) = A_c \sum_{k=-\infty}^{\infty} cos(2\pi f_c t + \theta_c + \varphi k) p(t - kT) \tag{2.1}$$

Hierbei bezeichnet A_c die Amplitude des sinusförmigen Signals und f_c stellt die Trägerfrequenz zum momentanen Zeitpunkt t in Sekunden dar. Die Phase des Trägersignals wird durch θ_c dargestellt, wobei die zu übertragende Information in φk enthalten ist. $p(t - kT)$ stellt die zeitliche begrenzte Impulsantwort dar. Um zwei binäre Werte (0/1) zu übertragen, wie es bei Binary Phase-Shift Keying (BPSK) der Fall ist, werden die beiden Phasenzustände des Trägers wie folgt dargestellt:

$$s_1(t) = A_c\, cos(2\pi f_c t) \qquad 0 \leq\ t \leq\ T_b \quad \text{für } binary\ 1 \tag{2.2}$$

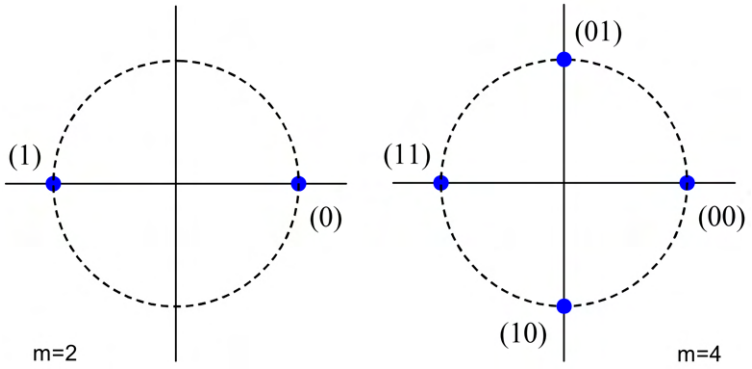


Abbildung 2.1: Die Signalraumkonstellation für BPSK mit $m = 2$ (links) und QPSK mit $m = 4$ (rechts) dargestellt. Die Angabe m bezeichnet die Anzahl der Informationen die übertragen werden (Eigene Darstellung, in Anlehnung an: [10]).

$$s_0(t) = A_c\, cos(2\pi f_c t + \pi) \quad 0 \leq\ t \leq\ T_b \quad f\ddot{u}r\ binary\ 0 \tag{2.3}$$

Das Signal $s_1(t)$ repräsentiert das Trägersignal für die Übertragung einer binären Informationseinheit mit dem Wert 1. $s_0(t)$ steht für die Übertragung des binären Wertes 0 wohingegen T_b steht für die Bitdauer in Sekunden bezeichnet. Bei der Übertragung von mehr als zwei Informationen, wie es beispielsweise bei QPSK der Fall ist, werden die Phasenwinkel 0, 90, 180 und 270 Grad für die Übertragung verwendet werden. Die Abbildung 2.1 zeigt die Phasenlagen der BPSK und QPSK Modulation mit konstanter Amplitude.

2.1.2 Amplitude-Shift Keying

Im Gegensatz zum PSK-Verfahren wird beim Amplitude-Shift Keying (ASK)-Verfahren die Amplitude des Trägersignals verändert, während die Phase des Trägersignals konstant bleibt. Für das modulierte Signal gilt

$$x_c(t) = A_c\, x_i(t) cos(2\pi f_c t + \theta_c) \tag{2.4}$$

Die zu übertragende Information wird durch die Quadraturkomponenten $x_i(t)$ zum Zeitpunkt t dargestellt. Abbildung 2.2 stellt die ASK-Signalraumkonstellation dar. Die Übertragung von zwei Informationen mit m=2 (0/1) und mit m=4 (00/01/10/11).

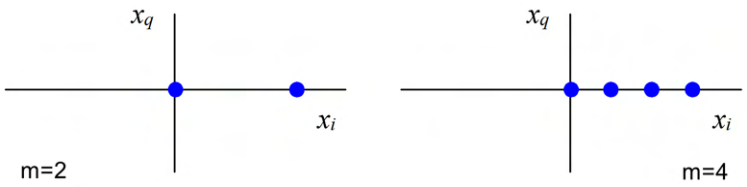


Abbildung 2.2: Darstellung der Signalraumkonstellation für ASK mit m=2 (links) und m=4 (rechts). m stellt die Anzahl der übertragenden Informationen, die auf der x-Achse als Real-Wert aufgetragen sind (Eigene Darstellung, in Anlehnung an: [10]).

2.1.3 Quadratur Amplituden Modulation

Die Quadratur Amplituden Modulation ist eine Kombination der beiden Modulationsverfahren ASK und PSK und nutzt Phase und Amplitude zur Informationsübertragung. Der Vorteil dieser Modulation liegt in der effizienten Nutzung der Bandbreite, da mit QAM eine hohe Informationsmenge pro Träger dargestellt werden kann. Nachteilig wirkt sich die Verwendung der Amplitudenmodulation aus, weil die Amplitude des Signal proportional zur Entfernung sinkt. Kompensiert wird dies indem eine Normierung der Amplitudenpegel zur Auswertung durchgeführt wird. Zudem ist aufgrund der Amplitudenänderung die Reichweite des Signal etwas geringer, als bei der Verwendung von PSK [11]. Der Grund dafür ist, dass die hohe Anzahl von Zeichen mit reduzierter Amplitude und damit geringerer Sendeleistung abgestrahlt wird.

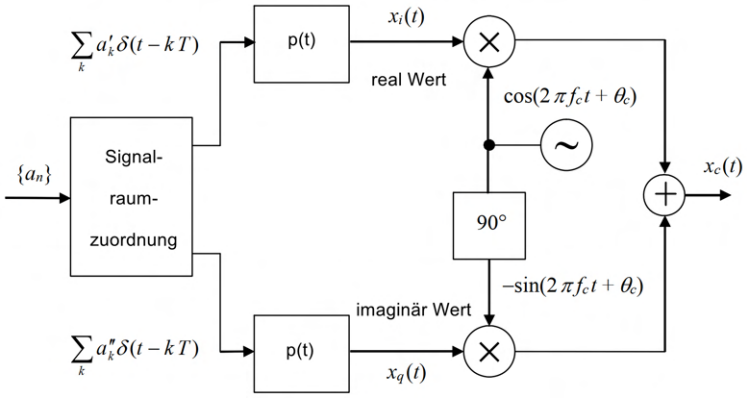


Abbildung 2.3: Quadratur-Modulator für die Erzeugung des QAM-Signals. In der Signalraumzuordnung erfolgt die 2D-Mapping des Informationssignal a_n. Die erzeugten Summensignale werden nach den Impulsformfilter $p(t)$ amplitudenmoduliert und aufsummiert. Nach der Additionsstufe liegt das QAM-Symbol $x_c(t)$ vor (Eigene Darstellung, in Anlehnung an: [10]).

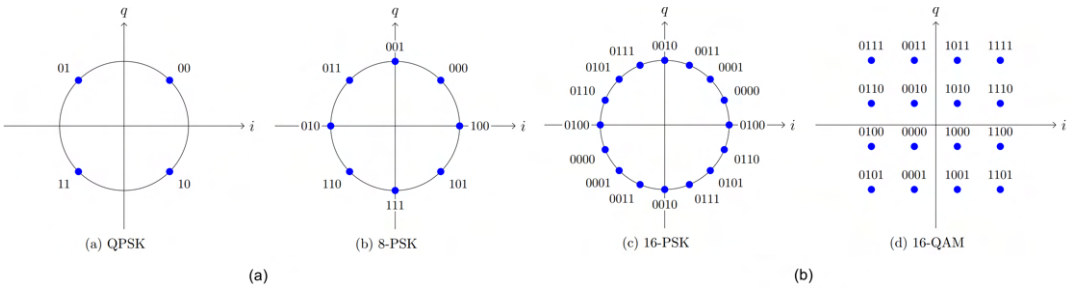


Abbildung 2.4: QPSK und QAM Modulationen im Vergleich mit der jeweiligen Angabe der übertragenden Information m. Die Achsenangabe q steht für den Imaginär- und i für den Realwert. Links (a) dargestellt die 4-QPSK(a) und die 8-PSK(b) Modulation. Die rechte Abbildung (b) zeigt den Unterschied zwischen der 16-PSK(c) und der 16-QAM(d) Modulation (Eigene Darstellung, in Anlehnung an: [12]).

Der QAM-Modulator (Abbildung 2.3) führt durch die Signalraumzuordnung eine 2D-Mapping des Datenstroms a_n durch. Die Summensignale werden dem Pulsformfilter zugeführt und dort als Quadraturkomponenten $x_1(t)$ und $x_q(t)$ ausgegebenen. Das Signal $x_i(t)$ wird mit $cos(2\pi fct + \Theta_c)$ amplitudenmoduliert. Dementsprechend wird $x_q(t)$ mit $-sin(2\pi fct + \Theta_c)$ moduliert. Die beiden Signale werden addiert und ergeben das QAM-Symbol $x_c(t)$.

In der Abbildung 2.4 sind QPSK und QAM im Vergleich der unterschiedlichen Datenübertragungsraten dargestellt. Die Modulation QPSK (a) ermöglicht die Übertragung von vier unterschiedlichen Zustände mit zwei Bits. Die Abbildungen (b) und (c) zeigen mit jeweils drei und vier Bits die ermöglichen Informationen, die übertragen werden können. Die Quadratur-Amplituden Modulation (QPSK) ist in (d) abgebildet.

2.2 Orthogonal Frequenz-Division Multiplexing

Das Orthogonal Frequency-Division Multiplexing (OFDM) [13] Verfahren wird verwendet um mit mehreren orthogonalen Trägern digitale Information zu übertragen. Eingeführt wurde dieses Multiplex-Verfahren mit dem 802.11a Standard im Jahr 2001.

2.2.1 OFDM-Symbol

Der Quadratur-Modulator (Abbildung 2.4) erzeugt QAM-Symbole, die durch Inverse Fast-Fourier-Transformation (IFFT) in orthogonale und überlappende Sinuskurven umgewandelt werden [14] [15]. Die IFFT Umwandlung in die OFDM-Symbole S_n kann mit Hilfe der Gleichung 2.5 dargestellt werden. $d(n)$ stellt dabei die Stichproben des Zeitindex (n) dar. m bezeichnet den Frequenzindex und N die Anzahl der

Samples, die eine Potenz von 2 sein müssen.

$$S_{(n)} = \frac{1}{N} \sum_{n=0}^{N-1} d(n) e^{i2\pi m \frac{n}{N}} \quad n = 0, .., N-1 \tag{2.5}$$

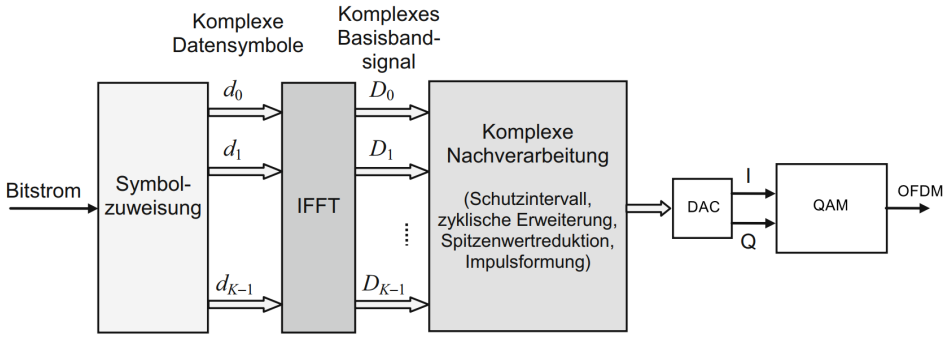


Abbildung 2.5: Umwandlung des Bitstroms in ein OFDM-Signal. Auf der linken Seite erfolgt die Einspeisung des Bitstroms in die Symbolzuweisung. Die erzeugten komplexen Datensymbole werden zur Weiterverarbeitung an den IFFT und zur Nachverarbeitung weitergegeben. Der Digital-Analog-Converter (DAC) leitet die Signale I und Q dem QAM zu und erzeugt das OFDM-Signal (Eigene Darstellung, in Anlehnung an: [12]).

Die Abbildung 2.5 zeigt den Verlauf der Datenströme zur Erzeugung der OFDM-Symbole. Der Bitstrom wird in der Symbolzuweisung zu komplexen Datensymbolen (d_0 .. d_{K-1}) mittels Quadraturamplitudenmodulation umgewandelt und anschließend der IFFT zugeleitet und als D_0 .. D_{K-1} ausgegeben.

2.2.2 Mehrträgerübertragung

Bei der digital kodierten Übertragung kann es aufgrund von aufeinanderfolgenden Symbolen zu einer Verzerrung des Signals kommen, die als Intersymbol Interferenz (ISI) bezeichnet wird. Diese wird durch die Bandbreitenbegrenzung des Kanals und durch Mehrwegeausbreitung wie Reflexionen oder Dispersion verursacht. Durch die ISI verursachten Überlagerungen ist eine fehlerfreie Datenübertragung der Daten nicht mehr möglich. Diese Störungen werden als Intersymbolinterferenz oder Symbolübersprechen bezeichnet und erschweren eine fehlerfreie Rückgewinnung der Informationen beim Empfänger. Symbole sind in diesem Zusammenhang Bezeichnungen für Daten oder Signale.

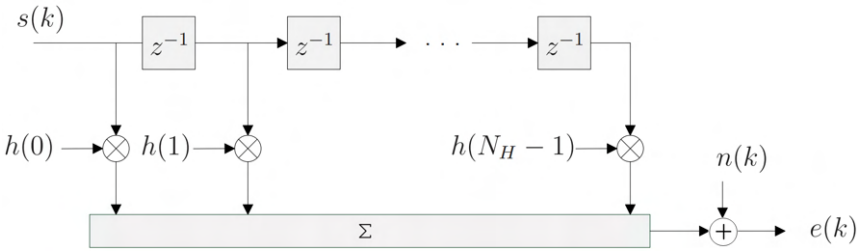


Abbildung 2.6: Modellhafte Darstellung eines frequenzselektiven Übertragungskanal. Das Sendesignal $s(k)$ mit den Verzögerungselementen z^{-1} wird mit den Gewichten $h(x)$ zum Summensignal aufaddiert. Das Empfangssignal $e(k)$ ergibt sich aus dem Rauschsignal $n(k)$ und den Summensignal. (Eigene Darstellung, in Anlehnung an: [16]).

Die Abbildung 2.6 zeigt das Modell eines frequenzselektiven Übertragungskanals. Hier werden die durch Mehrwegeausbreitung entstehenden Störungen durch eine endliche Länge dargestellt. $h(N_H - 1)$ stellt die maximal auftretende Verzögerung dar. Eine Beschreibung des Systems kann somit durch den Finite Impulse Response (FIR-Filter) mit der Ordnung N_H und den Koeffizienten $k(h)$ beschrieben werden [10]. Das Sendesignal $s(k)$ wird um die Elemente z^{-1} verzögert und mit den Koeffizienten $h(x)$ gewichtet. Das während der Übertragung auftretende Rauschen $n(k)$ wird am Ende zum Summensignal addiert.

Das daraus resultierende Empfangssignal $e(k)$ aus der Abbildung 2.6 kann durch die Gleichung 2.6 beschrieben werden und errechnet sich aus dem gefalteten Sendesignal $s(k)$ und dem Rauschen $n(h)$.

$$e(k) = h(k) \cdot s(k) = \sum_{k=0}^{N_H-1} h(k) \cdot s(k-n) = \sum_{i=0}^{N_H-1} h(i)s(k-i) + n(k) \tag{2.6}$$

Die Frequenzselektivität eines Übertragungskanals bewirkt, dass bestimmte Frequenzen bevorzugt oder benachteiligt werden. Dies kann sich nachteilig auf die weitere Signalverarbeitung auswirken. Mit Hilfe der Impulsantwort eines Übertragungssystems kann analysiert werden, ob es sich um Zeit- oder Frequenzdomäne handelt. Die Gleichung 2.7 beschreibt die Impulsantwort des Übertragungskanals.

$$H(f_i) = \sum_{i=0}^{N_H-1} h_i e^{-j2fi} = h_{fi} e^{-j\varphi f_i} \tag{2.7}$$

Aus der Gleichung 2.7 geht hervor, dass die Frequenz f_i um den Faktor h_{fi} gedämpft und um φf_i phasenverschoben ist. Die Faktoren h und φ sind abhängig von der jeweiligen Übertragungsfrequenz. Damit ist auch die Übertragungsfunktion des Signals frequenzabhängig. Aufgrund der Frequenzselektivität des Kanals ist beim Empfänger eine aufwändige Selektion der Empfangsfrequenz erforderlich.

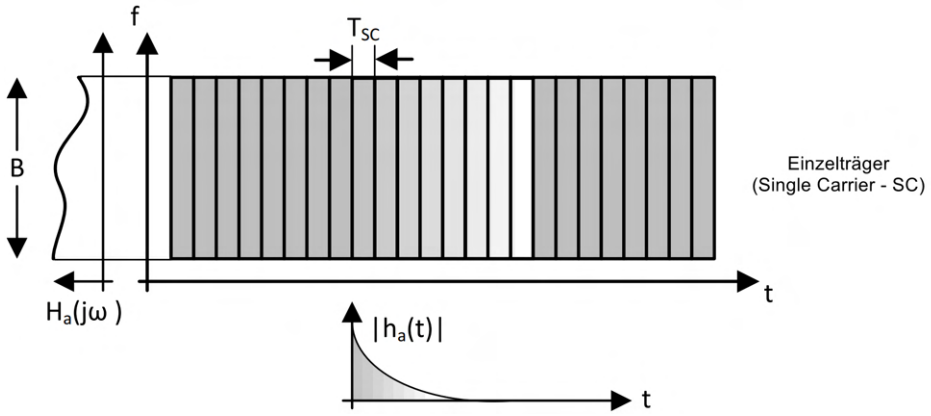


Abbildung 2.7: Darstellung des Einzelträgersignal (Single Carrier - SC) mit der Bandbreite B für die Frequenz f und dem Kanalfrequenzgang $H_a(j\omega)$. T_{SC} bescheibt die Symboldauer. Die Kanalimpulsantwort $|h_a(t)|$ wird unter dem Einzelträgersignal dargestellt (Eigene Darstellung, in Anlehnung an: [16]).

Die Abbildung 2.7 zeigt ein Einträgersignal, wobei T_{SC} die Symboldauer und $h_a(t)$ die Kanalimpulsantwort darstellt. Der Kanalfrequenzgang ist durch $H_a(j\omega)$ dargestellt. Bei der **fdm!** (**fdm!**)-Technologie werden anstelle eines Trägersignals mehrere Träger (Subträger) verwendet. Durch die Aufteilung des frequenzselektriven Kanals in schmalbandige Subkanäle sind diese nahezu unselektiv. Eine Korrektur des Kanaleinflusses im Empfänger wird dadurch wesentlich erheblich erleichtert.

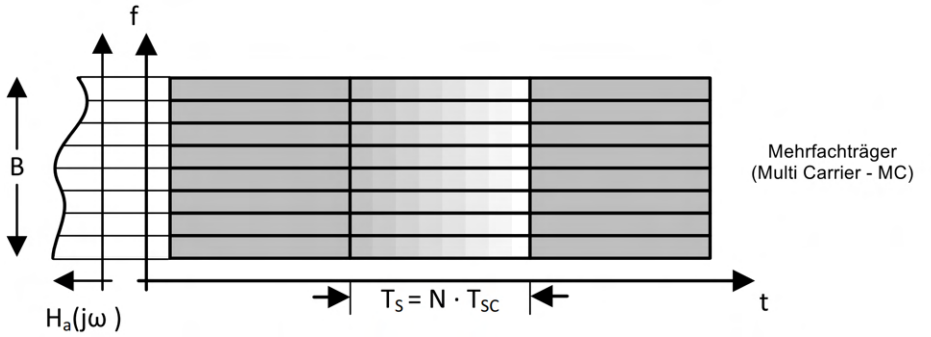


Abbildung 2.8: Spektrum des Mehrfachträgers. Die Symboldauer T_s berechnet sich aus der Faktor (Trägeranzahl) N und der Symboldauer TSC pro Träger (Eigene Darstellung, in Anlehnung an: [16]).

Die Abbildung 2.8 zeigt im Vergleich zur Abbildung 2.7, dass die Bandbreite des Signals nahezu identisch ist. Die Trägerbreite des Einzelträgersignals ist um den Faktor N größer als die des Mehrfachträgersignal. Im Gegenzug ist bei Mehrfachträgersignal die Symboldauer TS ist um den Faktor N größer als die des Einträgersignals. Das

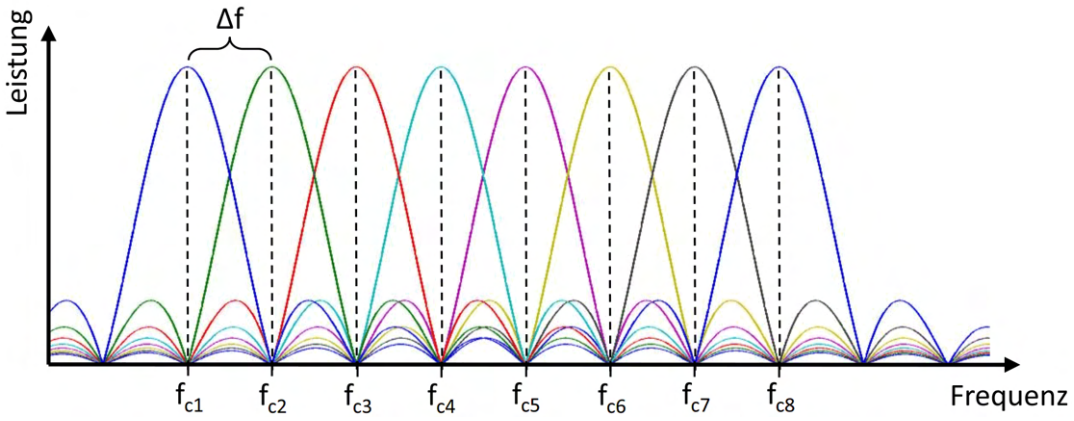


Abbildung 2.9: Das Leistungsdichtespektrum des OFDM-Signals mit den Trägern $f_{c1..8}$. Bei Erreichen des Leistungsmaximums eines Trägers befinden sich die jeweiligen anderen Träger im Nulldurchgang (Quelle: [16]).

Mehrträgersignal überdeckt lediglich einen Teil des Symbols, wodurch die Intersymbol Interferenz reduziert wird.

2.2.3 Orthogonale Träger

Im Gegensatz zur klassischen Datenübertragung, bei der nur ein Trägersignal verwendet wird, erfolgt beim Orthogonal Frequency-Division Multiplexing die Übertragung der Daten mit Hilfe von mehreren Unterträgern. Die Abbildung 2.9 zeigt das Spektrum eines OFDM-Signals. Die Orthogonalität der Subträger wird durch die Übereinstimmung des Maximums eines Subträgers mit den Nullstellen der anderen Subträger verdeutlicht.

Eine gleichmäßige Verteilung der Leistung auf alle Subträger K führt zu einer konstanten Leistungsdichte innerhalb der Bandbreite B (Gleichung 2.8). Der Subträgerabstand Δf ist mit einem Wert von 312,5 kHz ([10]) entsprechend dem WLAN IEEE 802.11ac Standard vorgegeben. Die Signaldauer wird mit T angegeben und ist durch den Subträgerabstand bestimmt.

$$B = K \cdot \Delta f = \frac{K}{T} \tag{2.8}$$

Ein Rechtecksignal (Informationssignal) enthält neben der Grundfrequenz auch Frequenzen oberhalb der Symbolrate. Bei der Übertragung dieses digitalen Signals über einen bandbreitenbegrenzten Kanal treten Verzerrungen auf. Diese äußern sich in Form von Verbreiterungen und werden als Intersymbol Interferenz bezeichnet [17] [18]. In einer Folge von digitalen Symbolen beeinflussen die Signalverbreiterungen die Abtastwerte der nachfolgenden Symbole, so dass diese nicht mehr korrekt zugeordnet werden können. Diesem Problem kann mit dem ersten Nyquist-Shannon-Abtasttheorem (Gleichung 2.9) begegnet werden. Das Theorem besagt, dass die Ab-

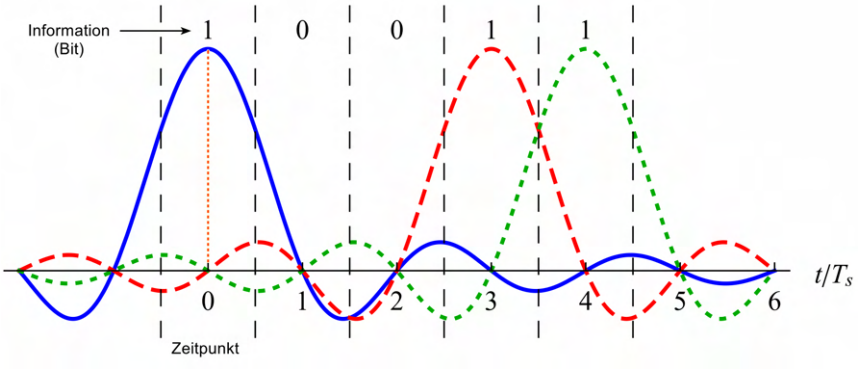


Abbildung 2.10: Darstellung eines Non-Return-to-Zero Signals mit den Entscheidungspunkten. Zum Zeitpunkt 0 erfolgt die Übertragung der Information 1 durch den Träger in blau. Zu diesem Zeitpunkt sind die roten und grünen Träger beim Nulldurchgang und haben somit den geringsten Einfluss auf das blaue Trägersignal (Eigene Darstellung, in Anlehnung an: [16]).

tastfrequenz f_{abtast} eines diskreten Signals mindestens doppelt so groß sein muss wie die höchste vorkommende diskrete Frequenz f_{max}.

$$f_{abtast} \geq 2 \cdot f_{max} \tag{2.9}$$

Die Abbildung 2.10 veranschaulicht das Prinzip der Vermeidung von Intersymbol Interferenz anhand eines unipolaren NRZ-Signals (Non-Return-to-Zero-Signals). Die Übertragung einer ”1” erfolgt zum Zeitpunkt $0, 3, 4$ wohingegen eine ”0” zum Zeitpunkt $1, 2$ übertragen wird. Die Abtastung des Signals im Empfänger erfolgt zu den Zeitpunkten $1, 2, ...$ im Abstand der Symboldauer T_s. Dadurch entstehen weder Nachläufer noch Intersymbol Interferenz, so dass die erste Nyquist-Bedingung erfüllt ist.

2.2.4 Schutzintervall und zyklische Erweiterung

Bei der Datenübertragung über einen frequenzselektiven Kanal wird das Empfangssignal durch ein Kanalecho überlagert, dessen Länge variieren kann. Nachfolgende Signale werden dadurch gestört [19]. Die Abbildung 2.11 zeigt die beim Ausschwingvorgang entstehenden Intersymbol Interferenz. Durch den Einfluss des Kanals treten während der Einschwingungsphase auch Intercarrier Interferenzen auf.

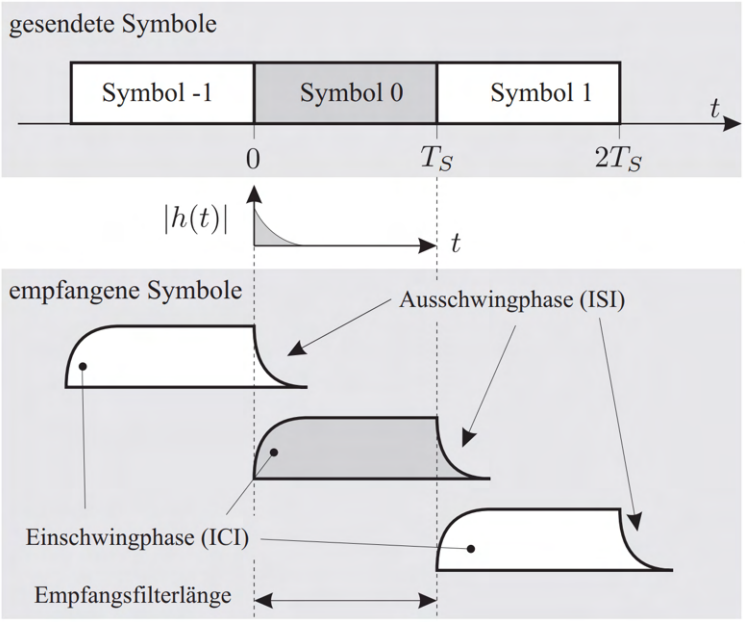


Abbildung 2.11: Symbole - Intersymbol- und Intercarrier-Interferenzen, die durch die Mehrwegeausbreitung entstehen. Die obere Abbildung zeigt die gesendeten Symbole. Die empfangenen Symbole werden durch die Ein- und Ausschwingvorgang beeinflusst (Quelle: [19])

Durch das Einfügen eines Schutzintervalls T_G können die Intersymbol Interferenz eliminiert werden. Damit ist die Intersymbol Interferenz-Freiheit wieder gewährleistet. Die Abbildung 2.12 zeigt die Verlängerung des Symbols um das Schutzintervall (T_G). Die Symboldauer T_S verlängert sich um den Wert TG. Dies hat zur Folge, dass sich die Symbolrate verringert. Der relative Durchsatz (D_{ref}) beträgt somit

$$D_{ref} = \frac{T_S}{T_G + T_S} \tag{2.10}$$

Nach der Erweiterung um den Schutzintervall TG beträgt die Symboldauer T_T (total-time):

$$T_T = T_G + T_S \tag{2.11}$$

Bei OFDM wird das Guardintervall durch eine zyklische Erweiterung des Kernsymbols realisiert, indem das letzte Stück des TG-Symbols der Länge T_G vor dem Beginn des Symbols hinzugefügt wird. Die Abbildung 2.12 verdeutlicht die Erweiterung um den Guard-Intervall T_G.

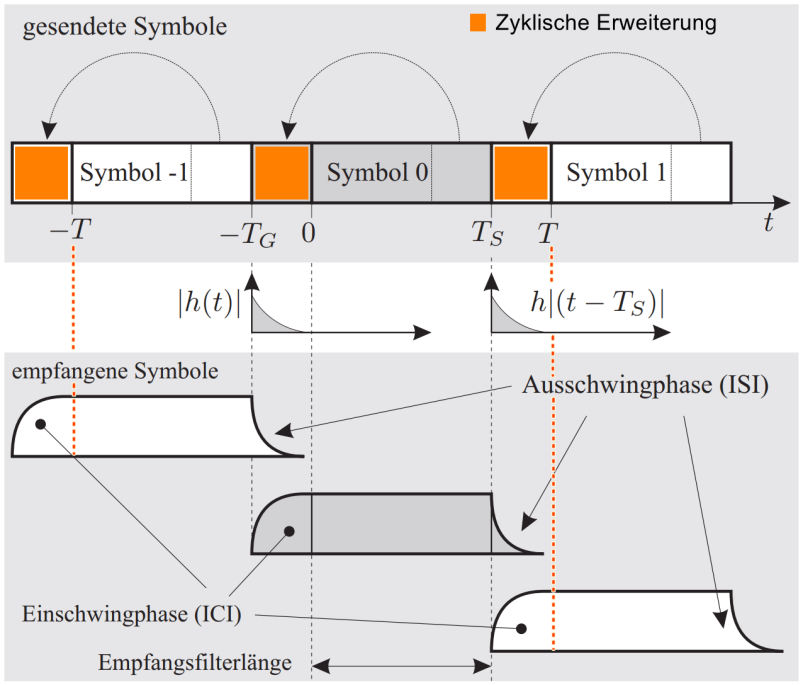


Abbildung 2.12: Durch die zyklische Erweiterung (farbliche Markierung) des Sende-Symbols mit dem Guard-Intervall können Interferenzen weitgehend eliminiert werden (Eigene Darstellung, in Anlehnung an: [19]).

2.2.5 Bandbreite

Der WLAN-Standard 802.11ac wurde 2013 von der IEEE spezifiziert. In der Tabelle 2.1 und Tabelle 2.2 sind die Kenndaten, sowie die Bandbreiten und Subträger für diesen Standard aufgeführt.

Tabelle 2.1: WLAN IEEE 802.11ac (Quelle: [17])

Modulationsverfahren	BPSK, QPSK, 16-, 64-, 256-QAM
Subträgerabstand Δf	312,5 kHz
Symboldauer T_s	3,2 μs
Schutzintervall (Cyclic Prefix) T_G	0,4 μs, 0,8 μs

Tabelle 2.2: 802.11ac Bandbreiten und Subträger (Quelle: [17])

Bandbreite	20 MHz	40 MHz	80 MHz	160 MHZ
Anzahl der Unterträger K	64	128	256	256
genutzte Subträger	56	114	242	484
Pilotträger	4	6	8	16

Die maximale Bitrate r_b für die Übertragung lässt sich daraus wie folgt berechnen:

$$r_b = log_2 m \; B \; R_c \; \frac{K_{Nutz}}{K} \; \frac{T_s}{T_G + T_s} \tag{2.12}$$

Die entscheidenden Größen für die Berechnung der maximalen Übertragungsbitrate sind das Modulationsverfahren (BPSK, QPSK, 16-, 64-, 256-QAM) und die Bandbreite. So können bei einer Bandbreite von 80 MHz und dem Modulationsverfahren 256-QAM maximal 292,5 Mbit/s übertragen werden.

$$6 \cdot 80 \cdot 10^6 \; bit/s \; \frac{5}{6} \; \frac{235}{256} \; \frac{3,2\mu s}{0,8\mu s + 3,2\mu s} = 292,5 \; Mbit/s \tag{2.13}$$

2.3 Mehrantennentechnik

Bei herkömmlichen Funksystemen erfolgt die Datenübertragung mit einer Sende- und einer Empfangsantenne. Dieses Verfahren wird als **siso!** (**siso!**) bezeichnet. Neuere Verfahren verwenden hingegen mehrere Antennen zum Senden und Empfangen. Die Verwendung mehrerer Sendeantennen und einer Empfangsantenne wird als Multiple Input Single Output (MISO) bezeichnet. **simo!** (**simo!**) hingegen bezeichnet die Verwendung von nur einer Sendeantenne und mehreren Empfangsantennen. Eine Kombination aus MISO und **simo!** wird als Multiple Input Multiple Output (MIMO) bezeichnet und stellt ein $N \times M$ Antennensystem [20] [21] dar. Der Einsatz von Mehrantennentechnik zielt darauf ab, die Redundanz und damit die Zuverlässigkeit des Signals beim Empfang zu erhöhen, die Bandbreite zu vergrößern und den Signal-Rausch-Abstand zu verbessern [20].

2.3.1 Multiple Input Single Output

Das MISO-Übertragungssystem verwendet mehrere Sendeantennen und eine Empfangsantenne. Die Abbildung 2.13 zeigt den Sender (Transeiver - Tx) und den Empfänger (Receiver - Rx). Der Sender verwendet das Sende-Diversity-Verfahren (Transmit Diversity) und das Space-Time Block Coding Verfahren um das Signal zu übertragen. Das Sende-Diversity-Verfahren ermöglicht es, mit mehreren Sendeantennen das gleiche Signal zu übertragen. Beim Space-Time Block Coding Verfahren, das auf dem Alamouti-Schema basiert, werden die zu übertragenden Daten in einem Raum-Zeit-Kodierer in Datenblöcke kodiert ([22]). Nach der seriell-parallelen Umsetzung der Datenblöcke werden die Symbole versetzt in Zeitschlitzen übertragen. [23]. Dieses Verfahren ermöglicht es, Rauschen und Fading (Signalpegelschwankungen) im Empfänger zu kompensieren und damit eine bessere Empfangsqualität zu erreichen.

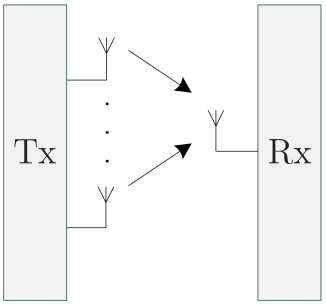


Abbildung 2.13: Dargestellt ist ein Multiple Input Single Output Übertragungssystem. Tx stellt den Sender (Transceiver - Tx) mit mehreren Antennen dar. Der Empfänger (Receiver - Rx) mit hingeben nur einer Empfangsantenne (Eigene Darstellung, in Anlehnung an: [20]).

2.3.2 Single Input Multiple Output

Beim **simo!**-Verfahren wird das Datensignals von nur einer Antenne gesendet, wohingegen der Empfang über mehrere Antennen erfolgt (Abbildung 2.14). Das Empfangssignal gelangt durch Reflexionen auf mehreren Wegen zum Empfänger. Diese Eigenschaft kann genutzt werden, um den bestmöglichen Empfang des Datensignals zu ermöglichen. Hierzu gibt es zwei unterschiedliche Ansätze. Zu einem das Switched Diversity und das Maximum Ratio Combining (MRC) Verfahren [22]. Beim Switched Diversity Verfahren wird immer das stärkere Signal vom Empfänger verwendet, während beim Maximum Ratio Combining (MRC) Verfahren das Summensignal genutzt wird. Beide Verfahren können im Empfänger einfach umgesetzt werden, da sie nur zwei HF-Pfade benötigen.

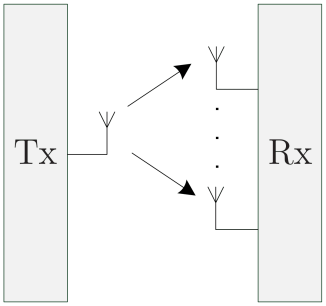


Abbildung 2.14: Abgebildet ist ein Single Input Multiple Output Übertragungssystem mit einem Sender (Tx) mit nur einer Antenne und das Empfangssystem (Rx) mit mehreren Empfangsantennen (Eigene Darstellung, in Anlehnung an: [20])

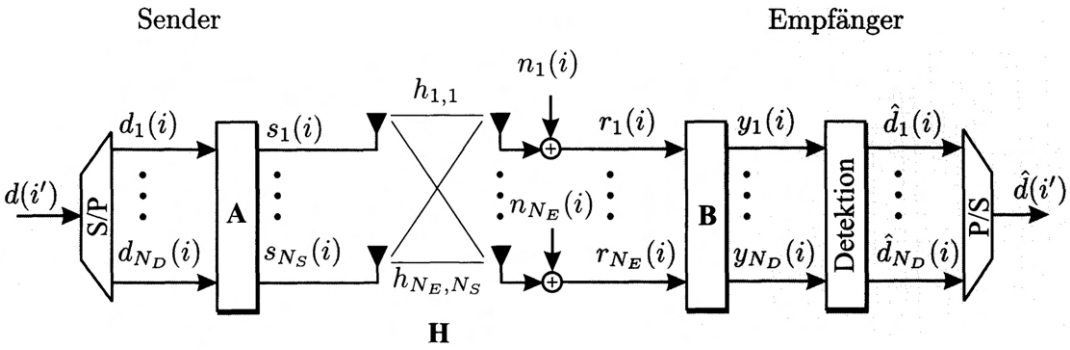


Abbildung 2.15: Das Modell eines Multiple-Input Multiple-Output Übertragungssystem (angelehnt an [20])

2.3.3 Multiple Input Multiple Output

Das Mehrantennensystem Multiple-Input Multiple-Output ist eine Kombination der beiden vorgenannten Systeme. Sowohl der Sender als auch der Empfänger verfügen über mehrere Antennen. Die Abbildung 2.15 zeigt das Modell eines MIMO-Übertragungsystems. Durch den Seriell/Parallel Umsetzer werden die Sendesymbole $d(i')$ zu den Vektoren $d(i)$ gruppiert und anschließend mit der Matrix A an den Sendevektor $s(i)$ weitergeleitet. Die Anzahl der Sendeantennen s wird durch N_S beschrieben. Am Empfänger erhält man N_e-dimensionale Empfangsvektoren $r(i)$. N_e beschreibt die Anzahl der Antennen am Empfänger. Nach der Multiplikation der Signale $r_{N_E}(i)$ erhält man nach der Matrix B die Vektoren $y(i)$ [22]. Das Empfangssignal kann als Matrix wie folgt dargestellt werden [23]:

$$\begin{bmatrix} r_1(i) \\ \vdots \\ r_{N_E}(i) \end{bmatrix} = \begin{bmatrix} h_{1,1} & \cdots & h_{1,N_S} \\ \vdots & \ddots & \vdots \\ h_{N_E,1} & \cdots & h_{N_E,N_S} \end{bmatrix} \cdot \begin{bmatrix} s_1(i) \\ \vdots \\ s_{N_S}(i) \end{bmatrix} + \begin{bmatrix} n_1(i) \\ \vdots \\ n_{N_E}(i) \end{bmatrix} \tag{2.14}$$

2.3.4 WLAN-Paket

Das in Unterabschnitt 2.2.1 beschriebene OFDM-Signal besteht aus mehreren Subträgern, die die Bandbreite des Signals bestimmen. Die Abbildung 2.16 zeigt den Zusammenhang zwischen OFDM-Symbolen, Subträger und Bandbreite. Für den WLAN-Standard 802.11ac wurde die Symboldauer T_s (Tabelle 2.1) mit $3,2\mu s$ festgelegt, woraus sich ein Subträgerabstand ($\Delta f = 1/T_s$) von 312,5 kHz ergibt. Bei einer Bandbreite von 20 MHz werden somit 64 Unterträger und 4 Pilotträger verwendet (Tabelle 2.1).

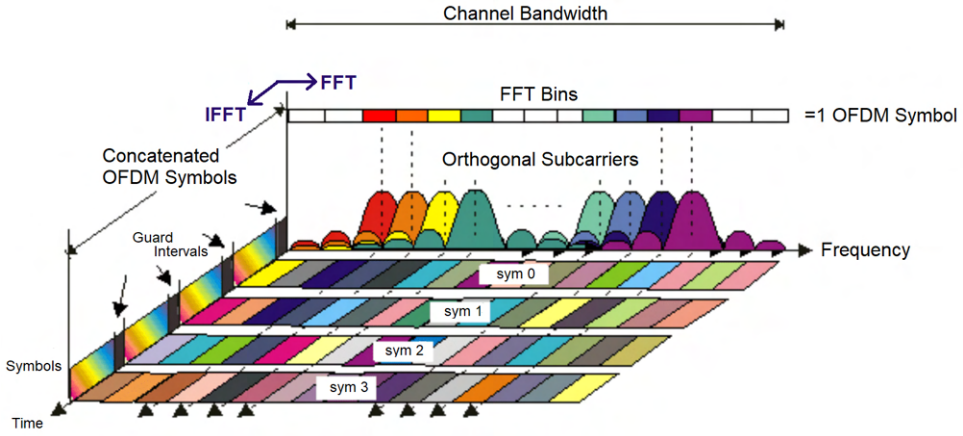


Abbildung 2.16: Die Frequenz-Zeit Darstellung eines OFDM-Signals zeigt die orthogonalen Subträger. Auf der Zeitachse abgebildet sind die Symbole mit den Guard-Intervallen zur Vermeidung der Intersignal- und Intercarrier-Interferenzen. Auf der X-Achse aufgetragen ist ein OFDM-Symbol. Die Y-Achse stellt jeweils ein Symbol dar (Quelle: [24]).

Die Abbildung 2.17 zeigt die Anordnung der Subträger für eine Bandbreite von 20 MHz. Die 54 Subträger verwenden eine Bandbreite von 16,875 MHz ($54 \cdot 312,5\ kHz$). Das Guardband verwendet die ersten sechs und die letzten fünf Nullträger für die analoge Filterung. Der mittlere Träger wird nicht verwendet, um DC-Offset-Probleme bei Direktmischempfängern zu vermeiden. Die vier Pilotträger dienen der Nachführung der Trägersynchronisation [15].

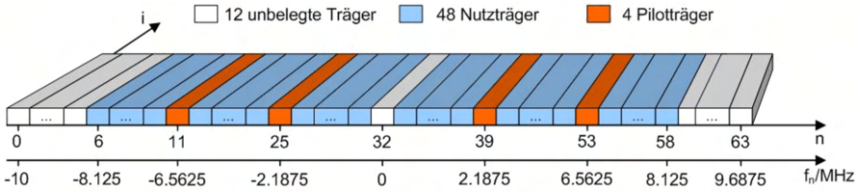


Abbildung 2.17: Die Abbildung stellt die Subträger-Anordnung bei IEEE 802.11a dar. Die Subträger 0 bis 5, 32 und 59 bis 63 werden nicht verwendet. Die Pilotträger 11, 25, 39 und 53 dienen zur Synchronisation. (Quelle: [15])

Für die Übertragung nach dem Standard IEEE 802.11a können verschiedene Übertragungsraten mittels **tdma!** (**tdma!**) verwendet werden. Die dabei übertragenen Rahmen (Bursts) bestehen aus mehreren OFDM-Symbolen. Jeder Burst beginnt mit einer Präambel und enthält die Grobsynchronisation, das **gi!** (**gi!**) sowie Referenzdaten (siehe Abbildung 2.18).

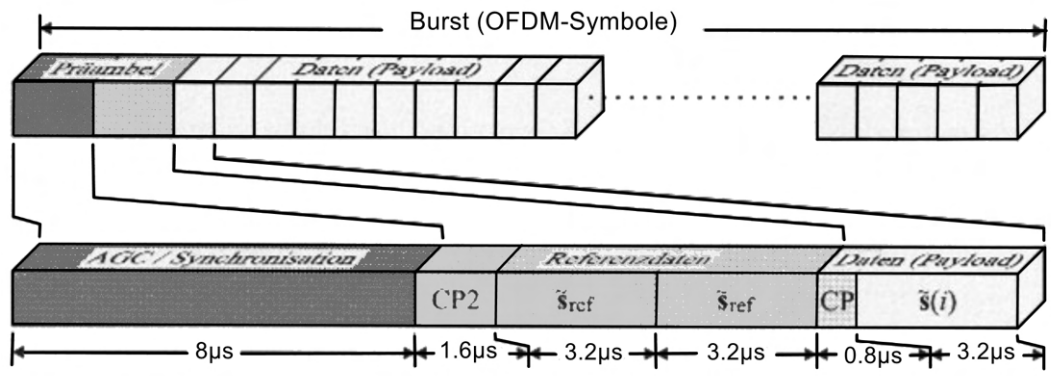


Abbildung 2.18: Die Darstellung der Burst-Struktur entsprechend des IEEE 802.11a Standards. Der Aufbau der Präamble und der Daten werden im entsprechenden Abschnitt detailliert aufgeführt. (Eigene Darstellung, in Anlehnung an: [15]).

Die Synchronisation benötigt incl. der adaptive Leistungskontrolle $8\mu s$ (Automatische Gain Control (AGC)). Danach folgt ein doppeltes Guardintervall von $1,6\mu s$, um Verzerrungen bei den nachfolgenden Trainingssymbolen zu vermeiden. Die Referenzdaten (Trainingssymbole) benötigen $2 \cdot 3,2\mu s$ und werden für die Feinsynchronisation und Kanalschätzung benötigt. Danach folgen die Nutzdaten mit $3,2\mu s$, denen jeweils ein **gi!** mit $0,8\mu s$ vorangestellt wird, so dass die Gesamtlänge der Daten (Payload) $4,0\mu s$ beträgt [15].

2.4 Kanalzustandinformationen

Die drahtlose Datenübertragung zwischen Sender und Empfänger erfolgt bei WLAN im Frequenzbereich von 2,4 und 5 GHz Bereich. Diese hochfrequenten Funkwellen breiten sich im Raum in alle Richtungen aus. Dadurch können beim Empfänger mehrere Signale ankommen, die sich überlagern und das Signal verfälschen. Die Signale werden an Gegenständen wie Wänden und Personen reflektiert oder es entstehen Streuungen (Scattering). Das empfangene Signal ändert aufgrund der unterschiedlichen Signallaufzeiten sowohl die Amplitude als auch die Phasenlage. Die Amplitudenänderung entsteht durch Dämpfungen auf dem Signalweg. Phasenverschiebungen hingeben ergeben sich durch Laufzeitunterschiede zwischen der Direktverbindung (line-of-sight) und Reflexion an Objekten.

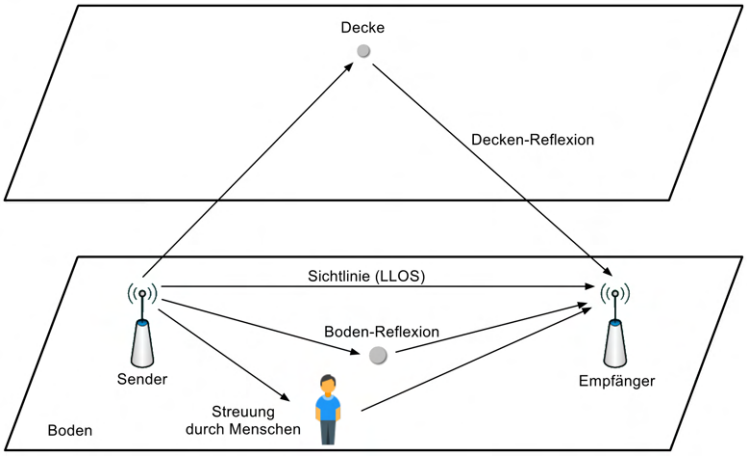


Abbildung 2.19: Die Darstellung zeigt die unterschiedlichen Reflexionen und Streuungen von Funkwellen die vom Empfänger zeitversetzt empfangen werden (Eigene Darstellung, in Anlehnung an: [25]).

Die Abbildung 2.19 zeigt die Reflexionen an Decke und Boden, sowie Streuung bei Personen. Um eine ausreichende Übertragungsqualität zu gewährleisten, wird bei OFDM das Kanalschätzverfahren verwendet. Hierzu überträgt der Sender Referenzsignale $t^{(k)}$ die beiden Teilnehmern (Sender und Empfänger) bekannt sind. Der Empfänger vergleicht die empfangenen Daten mit den Referenzwerten und nimmt die notwendigen Korrekturen vor. Dadurch wird die Übertragungsqualität optimiert und die Fehlerrate durch Minimierung der Kanalverzerrungen reduziert. Anschließend sendet der Empfänger Kanalzustandsinformationen an den Sender, um die notwendigen Anpassungen vorzunehmen [26]. Die in Abbildung 2.19 dargestellten Trainingssymbole werden hierbei zur Berechnung herangezogen. Wie in der Gleichung 2.14 dargestellt kann das empfangene Signal als

$$r^{(i)} = H^{(i)} \cdot s^{(i)} + n^{(i)} \tag{2.15}$$

dargestellt werden [27][28]. Der komplexe Vektor r wird gebildet aus der Matrix H welche die Kanalzustandsinformationen enthält, dem Signal s und dem Rauschen n. Pro Subunterträger enthält H eine Real- und Imaginär-Komponente. Die Unterträger werden durch i dargestellt. Der Empfänger berechnet mit Hilfe des empfangenen Signals $r^{(i)}$ und dem Referenzsignals $s^{(i)}$ die Kanalzustandsinformationen. Die Rauschkomponente $n^{(i)}$ kann nicht bestimmt werden und wird bei der Berechnung nicht berücksichtigt, sodass sich folgende Formel ergibt

$$H^{(i)} = \frac{r^{(i)}}{s^{(i)}} \tag{2.16}$$

Die Channel State Information $H^{(i)}$ beinhaltet komplexe Werte mit denen die Amplitude $A^{(i)}$ und die Phasenlage $\phi^{(i)}$ des Signals berechnet werden können. Verwendung findet hierzu die Gleichung 2.17 und Gleichung 2.18.

$$A^{(i)} = \sqrt{(h_{im}^{(i)})^2 + (h_r^{(i)})^2} \tag{2.17}$$

$$\phi^{(i)} = tan^{-1}(\frac{h_{im}^{(i)}}{h_r^{(i)}}) \tag{2.18}$$

2.5 Stand der Wissenschaft

Im Folgenden wird ein Überblick über bisherige Forschungsarbeiten zur Bewegungsdetektion mit Kanalzustandsinformationen gegeben.

Beispielsweise wurden Methoden zur Klassifizierung von physikalischen Objekten anhand von Kanalzustandsinformationen durchgeführt [29]. Die Aufzeichnung der CSI-Daten erfolgte mithilfe von zwei TL-WR842ND WLAN-Routern mit jeweils zwei Antennen. Die Klassifizierung der CSI-Daten erfolgte durch Machine-Learning-Algorithmen Decision Trees (DT), Support Vector Machine (SVM), k-Nearest Neighbors Methode (KNN) und Feed-Forwarded Neural Network (FFNN). Das Ergebnis zeigt, dass FFNN die besten Ergebnisse erzielt hat. Die Umsetzung gestaltet sich jedoch schwierig, da sorgfältig geeignete Parameter ausgewählt werden müssen.

Des Weiteren wurde die Überwachung von Sitzbelegungen in Räumen untersucht, um die Ausbreitung von Infektionskrankheiten zu minimieren [30]. Zur Klassifizierung der Sitzbelegungen wurden Convolutional Neural Network (CNN), sowie Machine-Learning-Algorithmen wie Naive Bayes (NB), Decision Trees, Support Vector Machine und k-Nearest Neighbors verwendet. Die durchgeführten Untersuchungen belegen, dass eine Klassifizierung der Sitzbelegungen mit einer Genauigkeit von bis zu 90 % möglich ist.

Weitere Untersuchungen befassen sich mit der Erkennung begrenzter Bewegungsrichtungen, kurzen Erkennungsabständen sowie der ungenauen Merkmalextraktion unter Einbezug von Kanalzustandsinformationen [31]. Hierbei werden mehrere Signalverarbeitungstechniken kombiniert, um Phasenverschiebungen und Rauschen zu eliminieren. Zu diesem Zweck erfolgt ein richtungsunabhängiges CSI-Fusions- und Sharing-Modell mit dem Namen CSI-F. Die kombinierten Verfahren basieren auf der Verwendung von neuronalen Netzen, insbesondere auf den sogenannten „Convolutional Neural Networks (CNN)“ und „Gated Recurrent Units (GRU)“, um die anfangs beschriebenen Probleme zu lösen.

Des Weiteren wurden diverse Methoden zur Erklärbarkeit und Interpretation von Machine-Learning-Methoden erörtert [32]. In den letzten Jahren wurden verschiedene Erklärungsmethoden entwickelt, die anhand unterschiedlicher Verfahren versuchen, kausale Abhängigkeiten zu erfassen. Die Qualität der Erklärungsmethoden sowie eine notwendige Mensch-KI-Schnittstelle die ein konzeptionelles Verständnis ermöglichen, werden laut der Studie in Zukunft noch eine wichtige Rolle spielen.

Eine weitere Studie befasst sich mit den Prinzipien des erklärbaren maschinellen Lernens sowie dessen praktischer Anwendung [33]. Im Rahmen dieser Studie wurde eine Umfrage durchgeführt, welche sich hauptsächlich auf datengesteuerte Methoden des maschinellen Lernens konzentrierte. Die Studie verfolgt das Ziel, Praktikern (m/w/d) aus der Industrie das Thema des erklärbaren maschinellen Lernens näherzubringen und sie bei der Anwendung adäquater Werkzeuge zu unterstützen.

3 Maschinelles Lernen

3.1 Einführung in Maschinelles Lernen

Das Europäische Parlament definiert den Begriff der Künstlichen Intelligenz wie folgt: „Künstliche Intelligenz ist die Fähigkeit einer Maschine, menschliche Fähigkeiten wie logisches Denken, Lernen, Planen und Kreativität zu imitieren. KI ermöglicht es technischen Systemen, ihre Umwelt wahrzunehmen, mit dem Wahrgenommenen umzugehen und Probleme zu lösen, um ein bestimmtes Ziel zu erreichen. Der Computer empfängt Daten (...), verarbeitet sie und reagiert. KI-Systeme sind in der Lage, ihr Handeln anzupassen, indem sie die Folgen früherer Aktionen analysieren und autonom arbeiten.“ ([34]). Der Begriff der künstlichen Intelligenz umfasst eine Vielzahl von Konzepten, darunter Machine-Learning, künstliche neuronale Netze und Deep Learning (Abbildung 3.1). Maschinelles Lernen bezeichnet die Fähigkeit eines Algorithmus, aus Daten zu lernen, Entscheidungen zu treffen oder Klassifizierungen vorzunehmen. Der Algorithmus wird dabei so erstellt, dass er aus einer Menge von Daten ein Ergebnis erzeugt. Die Fähigkeit, Muster in Daten zu erkennen oder eine entsprechende Approximation der Daten zu erstellen, wird durch Training mit einer Vielzahl von Daten erlernt. Neuronale Netze sind dem menschlichen Gehirn nachempfunden und bestehen aus einer Eingabeschicht, verborgenen Schichten und einer Ausgabeschicht. Im Gegensatz zu neuronalen Netzen zeichnet sich Deep Learning durch eine höhere Anzahl an Layern aus, die zur Verarbeitung komplexerer Daten eingesetzt werden. Im weiteren Verlauf der Arbeit wird das maschinelle Lernen näher betrachtet.

3.1.1 Überwachtes Lernen

Beim überwachten Lernen (Supervised Learning) liegen beschriftete Daten (gelabelte Daten) vor. Die Zuordnung eines Datenpunktes (Merkmal) zu einer Ausgabe (Label) ist dem Algorithmus bekannt. Das Label repräsentiert hierbei eine Klasse. Beispielsweise gehört ein Apfel oder eine Birne zur Klasse Obst. Die Trainingsdaten dienen dem Algorithmus dazu, die Zuordnung zwischen Merkmalen und Labels zu erlernen, um Vorhersagen über neue Daten treffen zu können, die nicht zum Trainingssatz gehören. Die Trainingsdaten stellen dabei eine Teilmenge des gesamten Datensatzes dar. Die Restmenge, auch als Testdaten bezeichnet, wird verwendet, um zu überprüfen, wie gut der Algorithmus trainiert wurde, die richtige Zuordnung zu finden. Die beiden grundlegenden Algorithmen des Supervised Learning sind Klassifikation und Regression. Bei der Klassifikation werden die Objekte zu einer Klasse zugeordnet. Die Zielgröße ist dabei die Klassenzugehörigkeit [36]. Die Regression liefert dagegen als

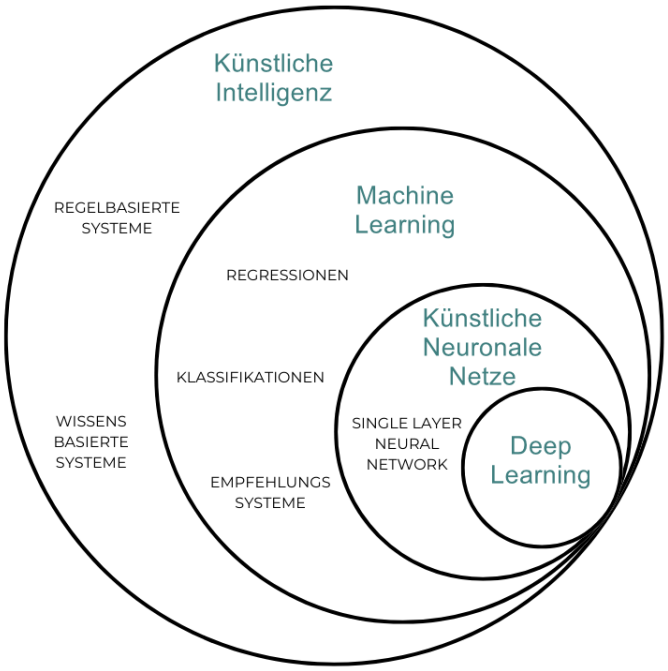


Abbildung 3.1: Die Darstellung zeigt die Hauptbereiche der Künstlichen Intelligenz als Oberbegriff für Machine Learning, Künstliche Neuronale Netze und Deep Learning. (Eigene Darstellung, in Anlehnung an: [35])

Zielgröße eine kontinuierliche Größe, die einen beliebigen Wertebereich haben kann. Das Ziel besteht in der Beschreibung des Zusammenhangs (Korrelation) zwischen abhängigen und unabhängigen Variablen in einem Datensatz mithilfe einer Funktion. Die lineare Regression findet Anwendung bei der Vorhersage zukünftiger Werte der abhängigen Variablen oder bei der Trendanalyse. Bei der binären Klassifikation erfolgt die Zuordnung eines Objekts zu einer von zwei Klassen. Im Falle einer Klassifikation von mehr als zwei Klassen wird von einer Multiclass-Klassifikation gesprochen. Charakteristisch hierfür ist ein Ziel (Label) mit mehr als zwei Kardinalitäten ([37]). In dieser Arbeit wird aufgrund der Datensets die Multiclass-Klassifikation verwendet.

3.1.2 Unüberwachtes Lernen

Beim unüberwachten Lernen (Unsupervised Learning) versucht der Algorithmus Ähnlichkeiten und damit Muster in den Daten zu erkennen, ohne vorher Informationen über die Zielgrößen zu haben. Der Algorithmus arbeitet dabei mit nicht gelabelten Daten, d. h. mit Daten, die keiner Kategorie (Label) zugeordnet sind. Das Ziel ist Zusammenhänge in den Daten zu finden. Der Algorithmus vergleicht die Daten miteinander und gruppiert sie zu Clustern. Unsupervised Learning wird bei Datensätzen eingesetzt, bei denen keine oder nur schwer zu beschaffende gelabelte Daten vorhanden sind.

3.1.3 Verstärktes Lernen

Verstärktes Lernen (Reinforcement Learning) basiert auf dem Prinzip von Versuch und Irrtum. Der Algorithmus (Agent) lernt durch Bestrafung oder Belohnung. Zu Beginn des iterativen Lernprozesses hat der Agent keine Informationen über seine Umgebung. Mit jedem weiteren Schritt lernt er seine Umgebung kennen und korrigiert seine Aktionen, um sein Ziel zu erreichen. Die Rückmeldung (Belohnung/-Bestrafung) veranlasst den Algorithmus, seine Strategie/sein Verhalten anzupassen. Reinforcement Learning findet Anwendung in unbekannten Umgebungen oder Situationen wie beispielsweise in der Robotik oder beim autonomen Fahren [36]. Der Agent initiiert eine Aktion in der Umgebung und erhält als Feedback eine Belohnung oder Bestrafung. Daraufhin passt er seine Strategie entsprechend an und führt erneut eine Aktion aus.

3.2 Maschine-Learning-Algorithmen

Im Folgenden werden geeignete Machine-Learning-Algorithmen von scikit-learn ausgewählt und beschrieben. Die Modelle wurden bislang noch nicht auf die zugrunde liegenden Daten beim BSI angewendet. Im Rahmen der Analyse der Kanalzustandsinformationen ist die Detektion von Bewegungen von zentraler Bedeutung. Die zur Verfügung gestellten Daten sind bereits mit entsprechenden Labels versehen. Die Analyse der Kanalzustandsinformationen erfolgt zur Erkennung von Bewegungen unter Anwendung von Methoden des überwachten Lernens (Unterabschnitt 3.1.1) um das Klassifizierungsproblem zu lösen. Im Rahmen der Vorverarbeitung (Preprocessing) der Datensets finden zudem Methoden Anwendung, die aus dem scikit-learn-Framework stammen.

3.2.1 Hyperparameteroptimierung

Im Rahmen des maschinellen Lernens werden Hyperparameter verwendet, um den Lernalgorithmus zu steuern. Im Gegensatz zu den Modellparametern, welche aus den Trainingsdaten erlernt werden, müssen die Hyperparameter vor dem Training des Modells vom Benutzer festgelegt werden. Diese sind für jedes Modell unterschiedlich und stellen einen Satz von Variablen dar. Die Bestimmung der Hyperparameter erfolgt experimentell, indem die Modelle eine Reihe von Ausprägungen desselben Hyperparameters durchlaufen, um den optimalen Wert zu finden. Bei der Auswahl der Hyperparameter gibt es verschiedene Verfahren zwischen denen unterschieden werden kann. Beim Gridsearch wird eine endliche Menge an Parametern angegeben, die als Grid über den Hyperparameter-Raum betrachtet werden. Der Algorithmus durchläuft alle Kombinationen, um die optimalen Werte zu finden. Je größer der Hyperparameter-Raum und komplexer die Modelle, desto höher ist der Rechenaufwand zur Berechnung der Parameter. Im Gegensatz zum Gridsearch erfolgt bei der Random Search eine zufällige Auswahl der Parameter aus einem definierten Wertebereich, die für das Training des Modells angewendet werden. Die Bewertung des Modells erfolgt, wie auch bei

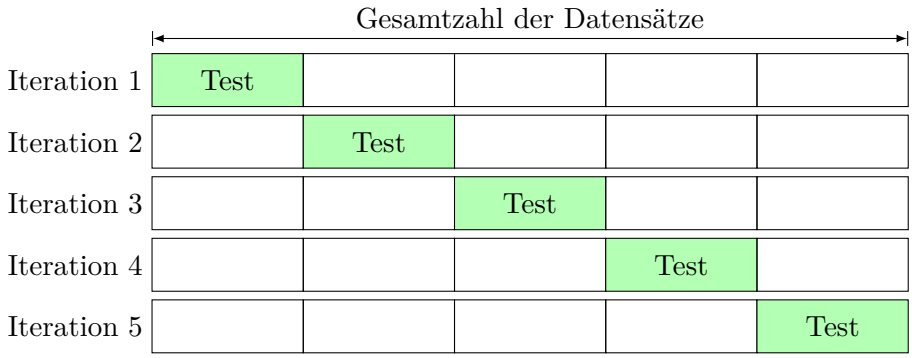


Abbildung 3.2: Darstellung der 5-fachen Kreuzvalidierung mit den entsprechenden Iterationen 1 bis 5. Die Aufteilung erfolgt dabei in fünf gleiche Datensets, wobei ein Datenset für das Training verwendet wird. Die Datensets 2 bis 5 werden für das Training verwendet. Nach jeder Iteration verschiebt sich das Testdatenset um ein Fold weiter. (Eigene Darstellung, in Anlehnung an: [38])

Grid Search, anhand der Testdaten. Die zufällige Auswahl der Hyperparameter kann dazu beitragen, dass RandomSearch schneller ein Optimum findet, wenn zwei Hyperparameter wenig korrelieren. Allerdings kann nicht sichergestellt werden, dass bei RandomSearch die optimalen Hyperparameter identifiziert werden, was im Gegensatz dazu bei GridSearch der Fall ist. Die Abbildung 3.3 zeigt den Vergleich zwischen Grid und RandomSearch. Wie auf der linken Seite ersichtlich, erfolgt eine systematische Suche aller Parameter, während auf der rechten Seite eine zufällige Auswahl der Parameter erfolgt. Um die Performance zu verbessern und das Overfitting-Problem zu reduzieren, verwenden GridSearchCV und RandomSearchCV die Kreuzvalidierung. GridSearchCV und RandomSearchCV bezeichnen bei scikit-learn die Funktion für die Rastersuche. Im Rahmen der Kreuzvalidierung erfolgt eine Aufteilung der Daten in Trainings- und Testdaten zu k großen Teilen. Der Algorithmus wird mit k-1 Folds trainiert und mit k-ten Folds validiert (Abbildung 3.2). Die Ergebnisse werden zur weiteren Verwendung abgespeichert. Der Trainings- und Validierungsprozess wird entsprechend der Aufteilung k-mal durchlaufen und abschließend das arithmetische Mittel aller Ergebnisse gebildet.

Die experimentelle Weiterentwicklung von GridSearchCV und RandomSearchCV stellt der HalvingGridSearchCV bzw. der HalvingRandomSearchCV Algorithmus von scikit-learn dar. Diese unterstützen eine schnellere Optimierung der Hyperparameter, indem eine sukzessive Halbierung der Suchstrategie verwendet wird. Anstatt die Parameter für die Hyperparameter unabhängig zu untersuchen, beginnen diese mit der Untersuchung aller Parameter jedoch mit nur einer geringen Menge an Ressourcen. Die schlechtesten Parameter werden nach jedem Durchlauf entfernt. Dadurch stehen mehr Ressourcen für die verbleibenden Parameter zur Verfügung. Dieser Vorgang wiederholt sich, bis nur noch ein Parameter übrig bleibt [39]. Obwohl die Nachteile,

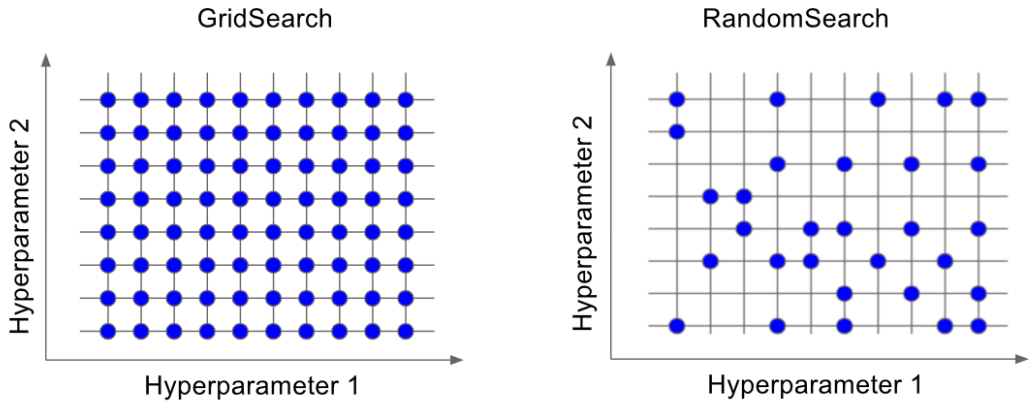


Abbildung 3.3: Die linke Darstellung zeigt GridSearch mit der systematischen Suche von zwei Hyperparametern 1 und 2. Rechts dargestellt RandomSearch mit einer zufälligen Auswahl der Parametern. (Eigene Darstellung, in Anlehnung an: [40])

wie das experimentelle Stadium der Funktion und keine Garantie für das Finden des Optimums bestehen wurde wegen der Vorteile HalvingGridSearchCV für die Arbeit verwendet. HalvingGridSearchCV wird aufgrund seiner Effizienz, Skalierbarkeit und Schonung von Ressourcen für die weitere Arbeit verwendet.

3.2.2 Support Vektor Machine

Die Support Vector Machine (SVM) gehört zur Klasse der überwachten Lernverfahren und wird zur Lösung von Klassifikations- und Regressionsaufgaben eingesetzt [41]. Das Ziel ist die Berechnung einer Hyperebene zwischen den Datenpunkten der verschiedenen Klassen, wobei der maximale Abstand (Margin) zwischen den Stützvektoren (Support Vektors) als Kriterium dient. Die Vektoren werden dabei so gewählt, dass sie nahe an den Klassengrenzen liegen [42]. Bei der Verwendung des Linear Support Vector Classifier (LinearSVC) stellt die Hyperebene eine Trennlinie dar (Abbildung 3.4).

Für nichtlineare Entscheidungsgrenzen wird der Kernel-Trick verwendet, der die Entscheidungsgrenzen in den höherdimensionalen Raum abbildet, ohne jedoch die Merkmale zu erweitern, so dass eine lineare Trennung möglich ist [44]. Die Abbildung 3.5 zeigt die Transformation in den höherdimensionalen Raum. Das Paket Scikit-learn unterstützt die gängigen Kernelfunktionen, darunter den linearen Kernel, den polynomialen Kernel, den radialen Basisfunktions-Kernel und den sigmoiden Kernel. Die Standardunterstützung für Support-Vektor-Maschinen beschränkt sich auf die binäre Klassifikation. Ein Ansatz zur Lösung des Multiklassifizierungsproblem mit Support Vektor Machine besteht in der Aufteilung des Datensatzes mit den Multiklassifikationen in mehrere binäre Klassifizierungsdatensätze. Für diese wird

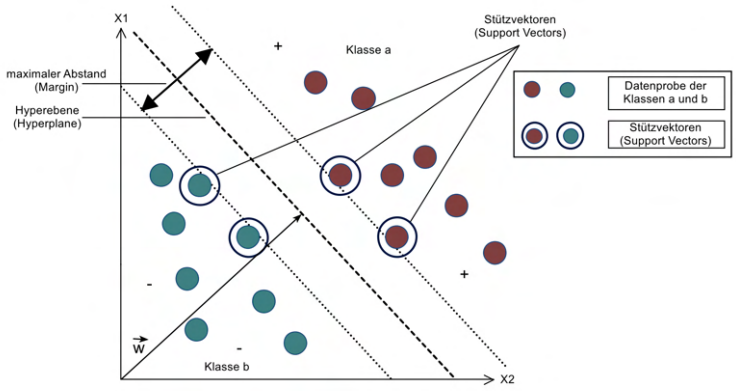


Abbildung 3.4: Die Abbildung zeigt die Hyperebene (Hyperplane) der linearen Support Vektor Machine zur Unterteilung der Klassen a und b. Die Stützvektoren (Supported Vektors) sind an den maximalen Abstand (Margin) angrenzend positioniert. Der Normalvektor $\vec{w}$ liegt orthogonal zur Trennlinie (Eigene Darstellung, in Anlehnung an: [43]).

die binäre Klassifizierung angewendet. Als Beispiel hierfür können die One-to-One- und die One-to-Rest-Strategie genannt werden. Bei der One-to-One-Strategie erfolgt die Klassifizierung in multiple binäre Klassifikationen. Im Rahmen dessen tritt jede Klasse gegen jede andere Klasse an. Demgegenüber erfolgt beim One-to-Rest-Ansatz eine Umwandlung in multiple binäre Klassifikation für jede Klasse ([45]).

3.2.3 Naive Bayes

Naive Bayes gehört zu den statischen Klassifikationsalgorithmen und basiert auf dem Bayes-Theorem. Das Bayes-Theorem besagt, mit welcher Wahrscheinlichkeit ein Ereignis eintritt und wird durch die Gleichung 3.1 dargestellt. $P(A|B)$ bezeichnet die Wahrscheinlichkeit, dass A eintritt, wenn das Ereignis B eingetreten ist [47]. Die Wahrscheinlichkeit, dass sowohl A als auch B eintreten, wird durch $P(A \cap B)$ ausgedrückt.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} = \frac{P(A \cap B)}{P(B)} \tag{3.1}$$

Das Ergebnis mit der höchsten Wahrscheinlichkeit wird als Kandidat betrachtet. Für den Klassifikationsalgorithmus lassen sich zwei wesentliche Annahmen formulieren [48]:

1. Die betrachteten Merkmale sind voneinander unabhängig und weisen keine gegenseitige Beeinflussung auf.

2. Die Bedeutung jedes Merkmals ist identisch.

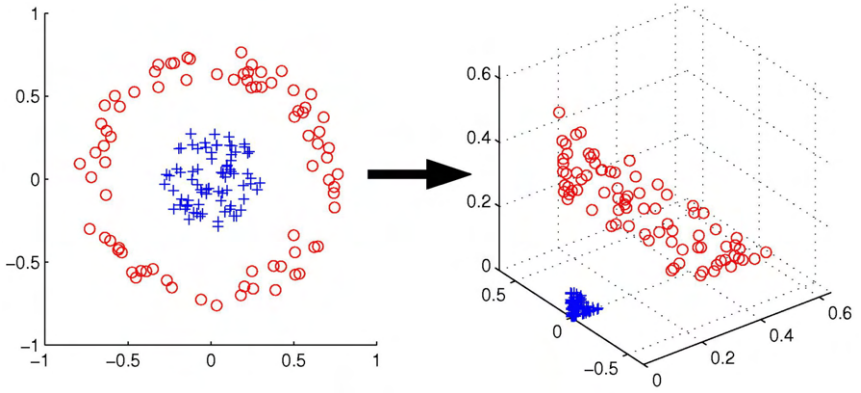


Abbildung 3.5: Die zweidimensionale Darstellung der Daten ist in der linken Abbildung dargestellt. Eine Klassifizierung der Daten mit Hilfe einer linearen Funktion ist nicht gegeben. Die Transformation in eine höhere Dimension mit Hilfe des Kernel-Tricks (rechts) ermöglicht hingegen die lineare Trennbarkeit der Daten (Quelle: [46]).

Naive Bayes Algorithmen lassen sich in verschiedene Kategorien einteilen, die mit scikit-learn implementiert werden können. Die am häufigsten verwendeten Algorithmen sind der Gaußsche, der Multinomiale und der Bernoulli Naive Bayes Algorithmus.

3.2.4 Logistic Regression

Eine der wichtigsten Klassifizierungstechniken des überwachten Lernens ist die Logistic Regression. Sie wird verwendet, um die Wahrscheinlichkeit des Auftretens dichotomer, als zweigliedriger Ereignisse vorherzusagen. Die am häufigsten verwendete Wahrscheinlichkeitsverteilung für zwei Klassen ist die Binomialverteilung [49]. Mit Hilfe der Maximum-Likelihood-Schätzung können die Parameter des logistischen Regressionsmodells geschätzt werden. Die Struktur des Modells besteht aus den zwei Funktionen f und g. f ist eine lineare Funktion, die stetige Werte (X) als Eingabeparameter verwendet und den Wert z als Ausgabewert liefert. Die Abbildung 3.6 zeigt die lineare Funktion f. x_n stellen die unabhängigen Variablen dar und die β_n Regressionskoeffizienten [50]. Die Werte von z werden an die nichtlineare Sigmoidfunktion g übergeben Abbildung 3.7. Der Wertebereich von z erstreckt sich von $[-\infty, +\infty]$. Der Wert von y liegt im Bereich zwischen $[0, 1]$.

Die Kombination der beiden Funktionen f und g ergibt die Formel der logistischen Regression (Gleichung 3.2). P gibt die Wahrscheinlichkeit an, mit der die abhängige Variable y den Wert 1 annimmt. Wie aus der Formel ersichtlich, bestimmen die Werte der Modellparameter β_0 und $\beta_n X_n$ die Wahrscheinlichkeit. Der Parameter n gibt die Anzahl der unabhängigen Variablen an. Zur Bestimmung der Modellparameter wird die Maximum-Likelihood-Methode angewendet.

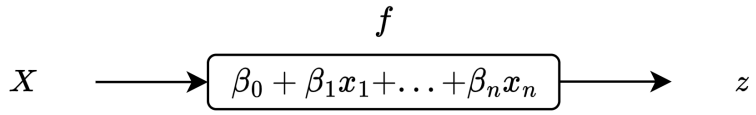


Abbildung 3.6: Die lineare Funktion der logistischen Regression lässt sich als gewichtete Summe der Eingangsmerkmale x_n und der Gewichte β_n darstellen (Eigene Darstellung, in Anlehnung an: [50])

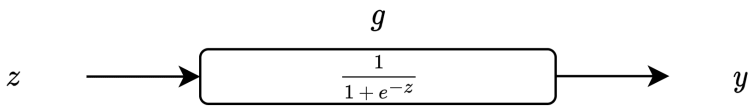


Abbildung 3.7: Die Sigmoid-Funktion g transformiert den Eingabewert z in eine Wahrscheinlichkeit (Eigene Darstellung, in Anlehnung an: [50])

$$P(y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n)}} \tag{3.2}$$

Die Likelihood-Methode beschreibt die Wahrscheinlichkeit, dass die beobachteten Daten durch die Kombination der Parameter $\beta_0, ..., \beta_n x_n$ generiert werden. Die Bezeichnung hierfür ist $L(\theta)$ [51]. Das Ziel der Maximum-Likelihood-Methode besteht nun darin, die Parameter $\beta_0, ..., \beta_n x_n$ zu finden, die die $L(\theta)$ Funktion maximieren. Dies kann mit der Log-Likelihood-Funktion geschehen. Für die Berechnung der Parameter gibt es verschiedene Methoden, wie beispielsweise die Fixed-Point-Iteration, die Bisection-Methode oder die Newton-Raphson-Methode [52], die jedoch nicht Gegenstand dieser Arbeit sind.

3.2.5 Decision Tree

Decision Trees (Entscheidungsbäume) weisen eine hierarchische Struktur auf und werden für Klassifikations- und Regressionsaufgaben verwendet [53]. Sie bestehen aus einem Wurzelknoten (root), Ästen, inneren Knoten und Blattknoten. Die Verzweigung der Äste erfolgt ausgehend vom Wurzelknoten zu den inneren Knoten. Die Entscheidung, ob eine Verzweigung zu einem inneren Knoten oder einem Blattknoten erfolgt, basiert auf den Merkmalen, die an den jeweiligen Knoten vorliegen (Abbildung 3.8). Als Ergebnis einer Klassifizierung repräsentieren die Blattknoten die Klassen. Im Falle einer Regression erfolgt die Schätzung der Zielvariable [54].

Zur Ermittlung der Homogenität der Klassifizierung in einem Entscheidungsbaum für einen spezifischen Punkt werden die Gini-Unreinheit (Gleichung 3.3) oder die Entropie (Gleichung 3.4) herangezogen [56]. Die Gini-Unreinheit G gibt die Gesamtvarianz über die Klassen K an. Der Anteil der Trainingsbeobachtungen der Knoten m in der Region k wird mit $\hat{p}_{mk}$ bezeichnet. Die Entropie D stellt eine Metrik zur Berechnung der Unreinheit der Verteilung dar [56].

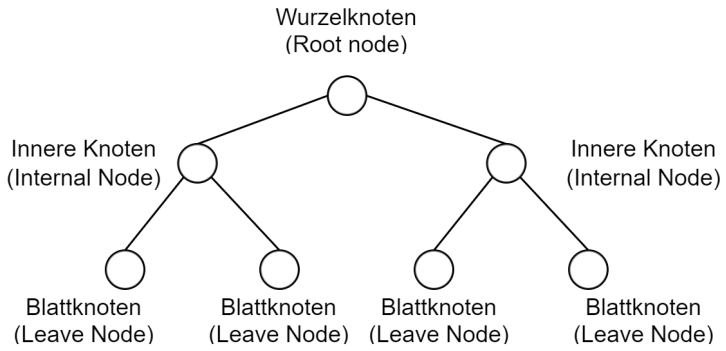


Abbildung 3.8: Entscheidungsbaum (Decision Tree) mit Wurzelkonten, inneren Knoten und Blattknoten. Die Ergebnisse der getroffenen Entscheidung werden durch diese Konten repräsentiert (Eigene Darstellung, in Anlehnung an: [55])

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{3.3}$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log \hat{p}_{mk} \tag{3.4}$$

3.2.6 Random Forest

Random Forest basiert auf einer Kombination von Entscheidungsbäumen. Hierbei werden mehrere Entscheidungsbäume trainiert und deren Vorhersagen kombiniert. Während bei Entscheidungsbäumen alle möglichen Merkmalsaufteilungen (Splits) berücksichtigen werden, wird bei Random Forest nur eine zufällige Teilmenge dieser Merkmale ausgewählt. Random Forest, der zu den Ensemble-Methoden (Unterabschnitt 3.2.7) zählen stellt, stellt eine Modifikation der Bagging-Methode dar, die im nachfolgenden Kapitel beschrieben wird. Bei der Klassifikation wird normalerweise die Mehrheit der Vorhersagen verwendet, um die Zugehörigkeit von Bäumen zu einer bestimmten Klasse zu bestimmen. Die Abbildung 3.9 zeigt einen Random Forest mit zwei identischen Klassen C.

3.2.7 Ensemble-Methoden

Ensemble-Methoden basieren auf Klassifikatoren wie beispielsweise Entscheidungsbäumen (Decision Trees). Sie kombinieren mehrere Algorithmen des maschinellen Lernens, um eine optimale Klassifikation oder Vorhersage zu erreichen. Ausgehend von schwächeren Lerner (weak lerners) [47] werden stärkere und genauere Modelle entwickelt, um Fehlerquellen wie beispielsweise die Verzerrung und die Abweichung zu verringern. Verzerrungen resultieren aus der falschen Annahme des Lernalgorithmus

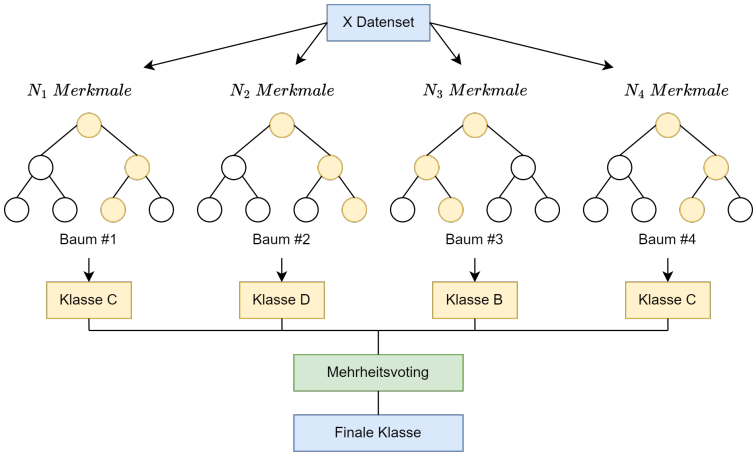


Abbildung 3.9: Die Abbildung zeigt den Random Forest, der auf Basis des Mehrheitsvotings die finale Klasse bestimmt. Die Aufteilung des Datensets erfolgt zufällig, welche anschließend in den Entscheidungsbäumen trainiert werden (Eigene Darstellung, in Anlehnung an: [57])

und können als Maß für die Abweichung der vorgesagten Werte und der wahren Werte der Funktion verstanden werden. Diese können zu einer Unteranpassung des Modells führen. Im Gegensatz dazu ist die Varianz das Maß der Abweichung des Vorhersagewerts vom Erwartungswert der Funktion. In diesem Fall erfolgt eine Überanpassung, bei der das Modell zu stark an die Trainingsdaten angepasst wird. Bei unbekannten Daten die Modelle eine schlechte Vorhersagegenauigkeit aufweisen. Der Punkt, an dem das Modell am besten generalisiert, wird als Bias-Varianz-Tradeoff bezeichnet. Die Abbildung 3.10 zeigt den Verlauf der Kurven für Unter- und Überanpassung.

In der Literatur finden sich zahlreiche Beispiele für häufig verwendete Ensemble-Modelle, darunter Bagging (Bootstrap Aggregation), Boosting und Stacking [49].

Das Bagging-Verfahren, auch als Feature-Bagging bezeichnet, verwendet verschiedene Teilmengen (Bootstraps) des Trainingssets auf einen Modelltyp an. Für jeden dieser Bootstraps werden Basismodelle erstellt und unabhängig voneinander ausgeführt. Die Teilergebnisse werden zu einem Gesamtergebnis aggregiert, entweder als Max-Voting oder durch Mittelwertbildung. Bagging-Ensembles sind folglich in der Lage, Schwankungen in den Trainingsdaten auszugleichen.

Beim Boosting erfolgt das Training von schwachen Lerner nacheinander. Diese bezeichnen Klassifikations- oder Regressionsmodelle, die nur geringfügig besser abschneiden als der Zufall. Dazu werden Teilmengen aus dem Datenset entnommen und das Modell damit trainiert. Die falsch vorhergesagten Stichproben dienen dem nachfolgenden Modell als Trainingsdatensatz. Dadurch verbessern die nachfolgenden Modelle die Vorhergehenden. Im Rahmen der sukzessiven Modellierung erfolgt eine Aggregation und Verbesserung der Ergebnisse.

Im Gegensatz zu Bagging und Boosting werden beim Stacking starke Lerner (strong learners) verwendet, um ein neues, robusteres Modell zu erzeugen. Als starke Lerner

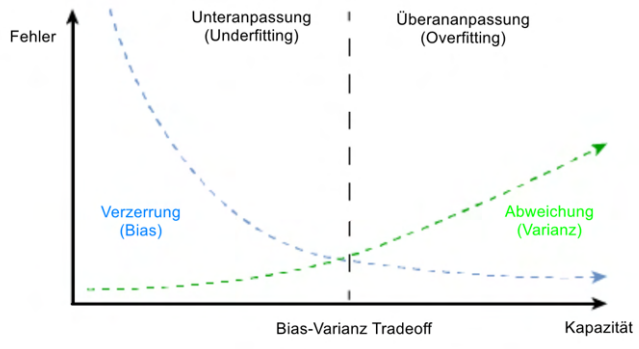


Abbildung 3.10: Der Schnittpunkt der beiden Kurven von Underfittung und Overfitting beschreibt den Zustand, bei dem das Modell am besten generalisiert wird. Dieser Zustand wird auch als Bias-Varianz-Tradeoff bezeichnet. (Eigene Darstellung, in Anlehnung an: [55]).

werden Modelle bezeichnet, die eine hohe Vorhersagegenauigkeit aufweisen. Der erste Schritt besteht darin, einen Trainingsdatensatz auf n starke Lerner anzuwenden. Die Ergebnisse dieses Trainingsdatensatzes werden zu einem neuen Trainingsdatensatz zusammengefasst. Auf Basis dieses Satzes wird ein Metamodell-Algorithmus erstellt, der die finalen Vorhersagen trifft. Im Vergleich zu Bagging und Boosting liefert Stacking präzisere Vorhersageergebnisse, ist rechenintensiver und benötigt mehr Zeit.

Zu den Ensemble-Methoden zähen neben dem Gradient Boosting auch der Extreme Gradient Boosting (XGBoost) Algorithmus. Dieser verfügt über algorithmische Optimierungen sowie die parallele Verarbeitung von Machine-Learning-Modellen [58]. Zur Optimierung des Algorithmus wird die sogenannte „pruning“-Methode verwendet, welche dazu dient, schwache Lerner (Entscheidungsbäume) zu kürzen oder gegebenenfalls komplett zu entfernen, bis sie optimal sind. Dadurch werden letztlich nur die besten Entscheidungsbäume berücksichtigt [59].

3.2.8 K-Nearest-Neighbor

Der Klassifikationsalgorithmus K-Nearest-Neighbor unterscheidet sich von den anderen Machine-Learning Klassifikationsalgorithmen dadurch, dass es sich um einen sogenannten Lazy-Learning-Algorithmus handelt. Dieser bildet die Modelle nicht anhand der Trainingsdaten, sondern speichert die Trainingsbeispiele. Die Vorhersagen werden durch Vergleich mit den gespeicherten Trainingsbeispielen getroffen sobald eine neue Anfrage eintrifft. Diese Vorgehensweise wird auch als instanz- oder speicherbasiertes Lernen bezeichnet [60]. Die Zuweisung von Datenpunkten erfolgt durch eine Mehrheitsentscheidung oder durch Gewichtung der Abstände. Dazu wird zunächst die Anzahl der Nachbarn k und das Abstandsmaß festgelegt. k bezeichnet die Anzahl der Nachbarn, die in die Bewertung einbezogen werden. Das Abstandsmaß gibt an, nach welcher Distance-Metric die Abstände des Datenpunktes k zu den Nachbarn be-

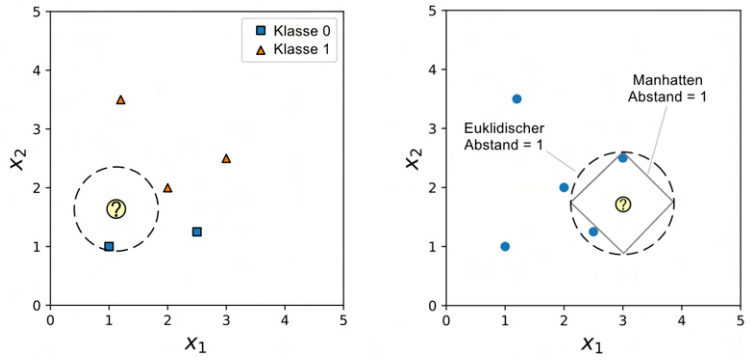


Abbildung 3.11: Der linke Bildabschnitt zeigt den zu klassifizierenden Datenpunkt, der mit dem gelben Fragezeichen gekennzeichnet ist. Die gestrichelte Linie stellt die Abstandsdistanz dar, in der sich der nächste Nachbar befindet. Die rechte Abbildung stellt den Unterschied zwischen der Euklidischen und der Manhatten Distanz dar. Die blauen Punkte stellen die Trainingsdaten dar. (Eigene Darstellung, in Anlehnung an: [61]).

rechnet werden. Hierzu wird der Abstand vom noch nicht klassifizierten Datenpunkt zu den nächsten Nachbarn berechnet. Der Algorithmus sucht nach k Exemplaren, die dem Datenpunkt am nächsten sind und entscheidet zu welcher Klasse dieser gehört. Zur Berechnung der Distanz kann je nach Anwendungsfall eine passende Abstandsfunktion verwendet werden. In der Praxis findet häufig die Euklidische Distanz (Gleichung 3.5) (auch als „Euklidischer Abstand“ bezeichnet) oder die Manhattan-Distanz (Gleichung 3.6) Anwendung. Die Distanz d zwischen den Punkten a und b in einem n-dimensionalen Raum wird durch diese beiden Distanzen beschrieben. Die K-Nearest-Neighbor-Verfahren veranschaulicht die Zuordnung zu einer Klasse in einem zweidimensionalen Raum.

$$d(a,b) = \sqrt{\sum_{i=0}^{n}(b_i - a_i)^2} \tag{3.5}$$

$$d(a,b) = \sum_{i=0}^{n}|b_i - a_i| \tag{3.6}$$

3.2.9 Confusion Matrix

Die Confusion Matrix ermöglicht die Bewertung der Güte von Modellen, indem sie eine Zusammenfassung der Vorhersageergebnisse für ein Klassifizierungsproblem darstellt. Sie stellt die Zuordnung von der tatsächlichen Klasse (Actual Class) und der

vorhergesagten Klasse (Predicted Class) dar. Der Wert True Positive und True Negative gibt denjenigen Wert an, bei dem die Vorhersage des Modells korrekt war. Ein False Positive bezeichnet eine fälschlicherweise positive Übereinstimmung mit einer Klasse, obwohl es sich in Wahrheit um eine negative Klasse handelt. Im Gegensatz dazu wird bei einem False Negative eine fälschlicherweise negative Zuordnung vorgenommen, obwohl es sich in Wahrheit um eine positive Zuordnung handelt. Die Abbildung 3.12 stellt die Confusion Matrix mit den beschriebenen Feldern in der blauen Umrandung dar. Die dargestellten Werte ermöglichen die Berechnung diverser Leistungsmaße der verwendeten Modelle. Die Precision gibt den Anteil der als positiv vorhergesagten Fälle an, die tatsächlich positiv sind. Der Recall-Wert beschreibt, wie viele der tatsächlich positiven Fälle als positiv vorhergesagt wurden unter Berücksichtigung der False-Negative-Rate. Relevante oder wichtige Ergebnisse werden durch einen hohen Recall-Wert dargestellt. Die Accuracy setzt die Anzahl der korrekt klassifizierten Fälle ins Verhältnis zur Gesamtzahl der Fälle und zeigt, wie gut das Klassifikationsmodell abschneidet. Der F1-Score (Gleichung 3.7) errechnet sich aus den Werten Precision und Recall unter Berücksichtigung die klassenbezogene Leistung des Modells.

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.7}$$

3.3 Vorverarbeitung der Daten

Der Datenvorverarbeitungsprozess spielt eine wesentliche Rolle beim maschinellen Lernen, da er die Leistung und Qualität der Modelle beeinflusst. Der Prozess stellt sicher, dass die Daten konsistent und für das Modell geeignet sind. Die Daten werden konvertiert und in eine Form gebracht, die für den Algorithmus des maschinellen Lernens optimal ist. Eine unzureichende Aufbereitung der Daten kann dazu führen, dass der Algorithmus falsche Ergebnisse liefert [47], da die Rohdaten fehlerhafte Formatierungen, inkonsistente, fehlende oder auch redundante Werte enthalten können. Das Ziel besteht somit in der Transformation der Rohdaten, sodass sie für den Algorithmus möglichst gut geeignet sind. Dies resultiert in einer gesteigerten Performance des Modells, einer verbesserten Interpretierbarkeit sowie einer erhöhten Robustheit [63].

3.3.1 Ausreißer

Innerhalb der Datenreihen lassen sich vereinzelt Datenpunkte identifizieren, die signifikant von den übrigen Daten abweichen. Durch thermisches Rauschen und Leistungsanpassung werden Ausreißer (Outlier) in das Signal eingebracht und sind als Anomalie in den Datenreihe erkennbar [4]. Die grafische Darstellung der Daten ermöglicht deren Visualisierung, beispielsweise mit mathplot oder seaborn. Zur Erkennung und Eliminierung von Ausreißern wird der Hampel-Filter verwendet. Dieser eignet sich insbesondere für Zeitreihen, um Ausreißer in den Datenreihen zu identifizieren

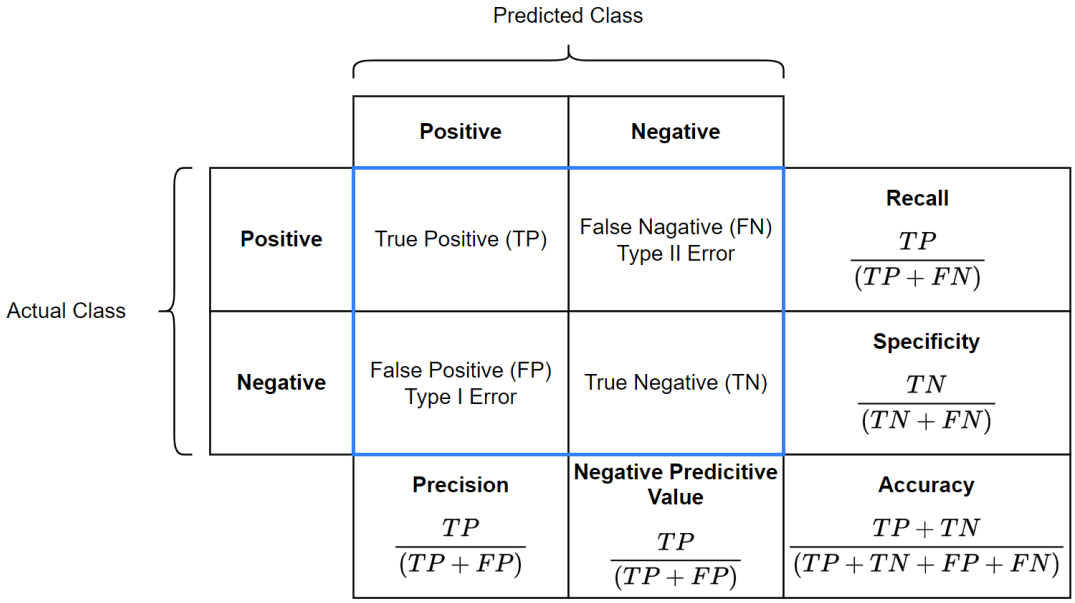


Abbildung 3.12: Die Abbildung zeigt links dargestellt die tatsächliche Klasse (Actual Class), oben die vorhergesagte Klasse (Predicted Class). Rechts und unten sind die Berechnungsformeln für die einzelnen Werte angeordnet. Der blau umrahmte quadratische Bereich in der Mitte dient zur Berechnung der Metriken. (Eigene Darstellung, in Anlehnung an: [62])

und zu korrigieren. Der Algorithmus analysiert die Datenreihe in einem definierbaren Bereich, der sich über die Datenreihe bewegt. Für dieses Fenster werden der Median und die Medianabweichung (MAD) berechnet. Ausreißer werden anhand des Vielfachen der MAD des Medians erkannt und ggf. geglättet. In diesem iterativen Prozess bewegt sich das Fenster weiter und führt Berechnungen und Anpassungen für die gesamte Datenreihe durch [64]. Um die Auswirkungen der Ausreißer-Entfernung auf die Klassifikation beurteilen zu können, werden die Machine-Learning-Algorithmen auf die Datensets mit und ohne Ausreißer angewendet und im Vergleich dargestellt. Die Abbildung 3.13 zeigt ein Datenset mit Ausreißern, die rechts entfernt wurden. Es existieren mehrere Methoden zur Entfernung von Ausreißer in den Datenset. Dazu zählen beispielsweise die z-score oder IQR-Methode. Diese werden in dieser Arbeit jedoch nicht betrachtet.

3.3.2 Standardisierung

Ein wesentlicher Schritt im Rahmen der Vorverarbeitung ist die Anpassung der Merkmale. Merkmale, auch als Features bezeichnet, stellen die Rohdaten numerisch dar und können sich auch auf deren Eigenschaften oder Variablen beziehen ([66]). Im weiteren Verlauf werden die Rohdaten als Daten benannt. Die Wertebereiche der

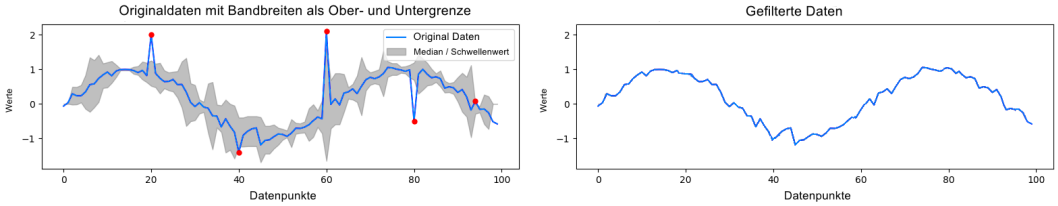


Abbildung 3.13: Links dargestellt die original Daten mit den Bandbreiten (grau) der Daten. Die roten Punkte kennzeichnen die Ausreißer. Die blaue Kurve stellt den Median der Datenpunkte dar. Die rechte Abbildung zeigt die Daten nach der Anwendung des Filters (Eigene Darstellung, in Anlehnung an: [65]).

Daten können erheblich variieren, was zu einer verminderten Leistungsfähigkeit der Maschine-Learing-Algorithmen und zu Fehlinterpretationen führen kann. Um dies zu vermeiden, ist es in der Regel notwendig, die Daten zu standardisieren. Scikit-learn stellt eine Reihe von Algorithmen zur Verfügung, die diese Aufgabe erfüllen. Für die Standardisierung von Daten, die einer Normalverteilung folgen, eignet sich der Standardscaler. Der Algorithmus passt die Merkmale derart an, dass der Mittelwert den Wert 0 annimmt und die Standardabweichung bei 1 liegt. Die Berechnung des skalierten Datenwertes X_{scaled} erfolgt nach der Gleichung 3.8.

$$X_{\text{scaled}} = \frac{x - \mu}{\sigma} \tag{3.8}$$

x bezeichnet den Originalwert und μ den Mittelwert der Daten. Die Standardabweichung wird durch σ angegeben, sodass Ausreißer die Berechnung verzerren können. Sind die Merkmale hingegen gleichverteilt, kann die Min-Max-Skalierung verwendet werden. Dabei werden die Merkmale angepasst, ohne ihre Verteilung zu verändern. Die Berechnung des skalierten Datenwertes X_{scaled} der Min-Max-Normierung zeigt die Gleichung 3.9.

$$X_{\text{scaled}} = \frac{X - min(X)}{max(X) - min(X)} \tag{3.9}$$

X stellt den ursprüngliche Wert des Merkmals dar. Der minimale und der maximale Wert von X sind durch $min(X)$ und $max(X)$ gegeben. Der RobustScaler-Algorithmus ist im Gegensatz zum MinMaxScaler robust gegenüber Ausreißern indem, er den Median entfernt und die Daten entsprechend des Quantilsbereich skaliert. Der Wert des Interquartilsbereichs (IQR) liegt standardmäßig zwischen dem 1.Quantil und dem 3. Quantil [67]. Die Berechnung des skalierten Wertes zeigt die Gleichung 3.10. Die Abbildung 3.14 zeigt die beschriebenen Scaler im Vergleich.

$$X_{\text{scaled}} = \frac{X - median(X)}{IQR(X)} \tag{3.10}$$

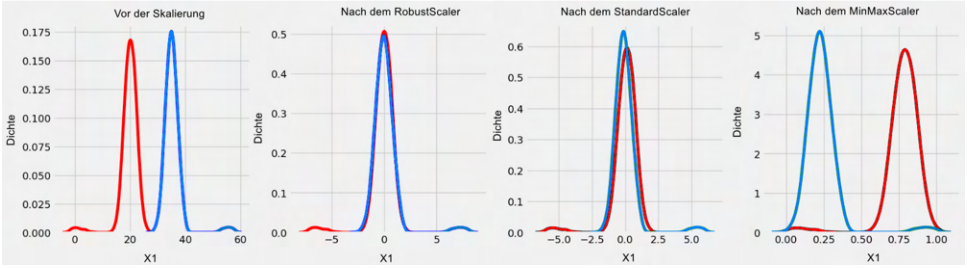


Abbildung 3.14: Die Abbildung zeigt die unterschiedlichen Ergebnisse des RobustScaler, StandardScaler und MinMaxScaler bei der Anwendung die Daten. Die rote und blaue Kurve veranschaulichen die jeweilige Dichteverteilung der Daten. Die Resultate des RobustScaler und StandardScaler differieren nur gering, wobei der Mittelwert bei Null liegt. Demgegenüber wird beim MinMaxScaler eine Skalierung zwischen 0 und 1 durchgeführt. (Eigene Darstellung, in Anlehnung an: [68]).

3.3.3 Normalisierung

Im Gegensatz zur Standardisierung, die auf den gesamten Datensatz angewendet wird, wirkt sich die Normalisierung nur auf eine Stichprobe aus. Dabei wird eine Komponente, die nicht Null ist, so umskaliert, dass ihre Norm (L1 oder L2) gleich Eins ist. Es erfolgt dabei keine Entfernung des Mittelwerts oder Skalierung nach Abweichung. Vielmehr wird eine Stichprobe auf einen Einheitsstandard normiert. Die Normierung kann hierzu auf drei verschiedene Arten erfolgen [69] [70]. x stellt hierbei den zu normierenden Wert dar.

1. L1-Normierung: Hierbei werden die Daten so skaliert, dass die Summe der absoluten Werte in einer Spalte den Wert 1 ergibt (Gleichung 3.11).
2. L2-Normierung: Die Normierung der Daten erfolgt so, dass die Summe der Quadrate in jeder Zeile den Wert 1 ergibt (Gleichung 3.12).
3. Max-Normierung: Die Skalierung der Daten wir auf Basis des Maximalwertes durchgeführt (Gleichung 3.13).

$$L1 = \sum_{i=1}^{n} |x_i| \tag{3.11}$$

$$L2 = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{3.12}$$

$$Max = |x_i| \tag{3.13}$$

3.3.4 Dimensionsreduktion

Die Dimensionen werden durch die Anzahl der Merkmale (Feature) repräsentiert und beeinflussen somit die Berechnung des Modells. Das Ziel besteht darin, vor der Modellierung des Modells die Dimension des Datensatzes durch geeignete Techniken zu reduzieren, um auf diese Weise die Rechenzeit des Maschine-Learing-Algorithmen zu verkürzen und bessere Ergebnisse zu erzielen. Zur Dimensionsreduktion stellt scikit-learn verschiedene Techniken zur Verfügung. Die im Folgenden vorgestellten Methoden beziehen sich hauptsächlich auf die in dieser Arbeit vorgestellten Klassifikationsalgorithmen.

Zur Dimensionalitätsreduktion wird häufig die Hauptkomponentenanalyse (Principal Component Analysis – PCA) eingesetzt. Die Idee hinter dieser Methode ist, dass mehrere Merkmale korrelieren und zu weniger Hauptkomponenten zusammengefasst werden, ohne die Aussagekraft des Datensatzes zu beeinträchtigen [47]. Durch die Überführung der Merkmale in einen neuen Vektorraum können die mehrdimensionalen Merkmale minimiert werden. Der neu gebildete Raum soll dabei möglichst wenige korrelierende Features (Principal Components) enthalten [71]. Dazu werden die Beziehungen (Kovarianzen) der Variablen aller Spalten zueinander berechnet. Die Berechnung der Eigenvektoren und Eigenwerte erfolgt auf Basis der in der Kovarianzmatrix gespeicherten Werte. Die Hauptkomponente der Daten wird durch die Eigenvektoren repräsentiert. Die Eigenwerte stellen die Varianz dar, welche durch die Hauptkomponente erklärt wird. Die größten Eigenwerte werden als die Hauptkomponente ausgewählt. Durch die Darstellung der Daten mittels der ersten Hauptkomponente kann die Dimension des Datensatzes stark reduziert werden, ohne dass wesentliche Informationen verloren gehen [72] (Abbildung 3.15).

Die Lineare Diskriminanzanalyse (LDA) kann wie die PCA zur Dimensionsreduktion verwendet werden. Im Gegensatz zur PCA wird bei der LDA die Klassenzugehörigkeit und nicht die Varianz zur Transformation verwendet. Dabei wird die Streuung zwischen den verschiedenen Klassen maximiert und die Streuung innerhalb der gleichen Klassen minimiert. Die Dimensionsreduktion erfolgt in mehreren Schritten. Im ersten Schritt wird die Varianz S_b (Gleichung 3.14) zwischen den Klassen g ($g \in 1, 2, ..., K$) berechnet. Dabei stellt N_i ($N \in \mathbb{N}$) den Stichprobenmumfang, $\overline{x}_i$ ($x \in \mathbb{R}$) den Gesamtmittelwert und $\overline{x}$ ($x \in \mathbb{R}$) den Mittelwert der jeweiligen Klasse dar [74].

$$S_b = \sum_{i=1}^{g} N_i(\overline{x}_i - \overline{x})(\overline{x}_i - \overline{x})^T \tag{3.14}$$

Der Abstand S_w zwischen Mittelwert und Stichprobe jeder Klasse wird gemäß der Gleichung 3.15 berechnet.

$$S_w = \sum_{i=1}^{g}(N_i - 1)S_i = \sum_{i=1}^{g}\sum_{j=1}^{N_i}(x_{i,j} - \overline{x}_i)(x_{i,j} - \overline{x}_i)^T \tag{3.15}$$

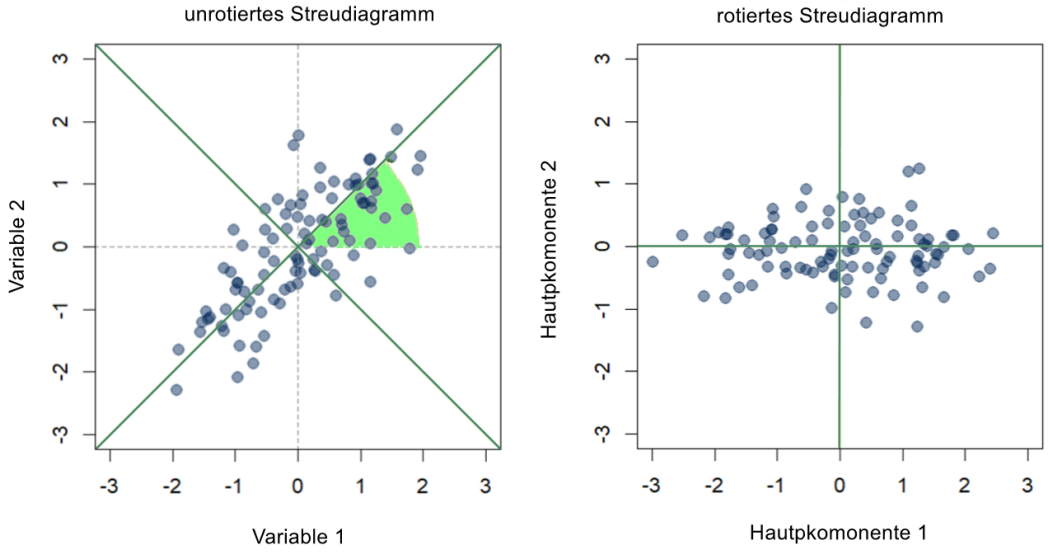


Abbildung 3.15: Das linke Diagramm zeigt die Hauptkomponente, welche durch die Streuwolke verläuft. Orthogonal hierzu ist die zweite Hauptkomponente dargestellt, welche den Anteil der Varianz in den Daten erklärt. Das rotierte Streudiagramm ist auf der rechten Seite des Diagramms dargestellt und zeigt die Hauptkomponenten im neuen Koordinatensystem (Eigene Darstellung, in Anlehnung an: [73]).

Schließlich wird ein niedrigdimensionaler Raum P_{lda} konstruiert, dessen Ziel es ist, die Varianz zwischen den Klassen zu maximieren und die Varianz innerhalb der Klassen zu minimieren [75] [76].

$$P_{lda} = arg_p max \frac{|P^T S_b P|}{|P^T S_w P|} \tag{3.16}$$

4 Methoden der Erklärbarkeit

Künstliche Intelligenz ist in der Lage, vielschichtige Probleme zu lösen. Die Entscheidungen, die zu diesen Lösungen führen, sind jedoch bei komplexeren Algorithmen nur schwer nachvollziehbar. Um die Entscheidungsfindung von KI-Systemen transparenter zu machen, gibt es den Ansatz der Erklärbaren Künstlichen Intelligenz (Explainable Artificial Intelligence, XAI). Explainable Artificial Intelligence versucht, die Entscheidungen von Algorithmen nachvollziehbar und damit vertrauenswürdig zu machen. Sie soll helfen zu erklären, warum maschinelle Lernmodelle zu den Ergebnissen kommen und auf welche Weise dies geschieht. Die *Ethics guidelines for trusworthy* [77] beschreiben diese wie folgt: „die Interpretierbarkeit bezieht sich auf eine passive Eigenschaft eines Modells, die sich auf die Ebene bezieht, auf der ein bestimmtes Modell für einen menschlichen Beobachter Sinn ergibt“ und „die Erklärbarkeit als aktives Merkmal eines Modells betrachtet werden, das jede Aktion oder Prozedur bezeichnet, die von einem Modell mit der Absicht durchgeführt wird, seine internen Funktionen zu verdeutlichen oder detailliert darzustellen“. In der Literatur wird der Begriff der Erklärbarkeit häufig synonym mit dem Begriff Interpretierbarkeit verwendet. Im weiteren Verlauf der Arbeit wird daher zur Vereinheitlichung der Begriffe Erklärbarkeit benutzt. Zudem erfolgt eine Klassifikationen von Modellen des maschinellen Lernens. Eine gängige Unterscheidung ist die Einteilung in das Blackbox- und Whitebox-Modell. Bei einem Blackbox-Modell ist die interne Struktur weitgehend unbekannt und die Entscheidungen des Modells sind nicht nachvollziehbar. Die Performance und Genauigkeit dieser Modelle ist jedoch als vorteilhaft zu bewerten. Im Gegensatz hierzu ist beim Whitebox-Modell die interne Struktur bekannt und die Entscheidungen des Modells sind nachvollziehbar. Dies wird in der Regel durch die Verwendung von linearen Modellen erreicht. Die bessere Interpretierbarkeit der Ergebnisse wirkt sich negativ auf die Leistungsfähigkeit des Modells aus.

In der Literatur finden sich verschiedene Methoden und Konzepte zur Erklärbarkeit von Modellen ([78]). Im Rahmen dieser Untersuchung wurde die visuelle Darstellung der Ergebnisse sowie die lokale und globale Erklärbarkeit des maschinellen Lernens verwendet. Die grafische Darstellung ermöglicht die Struktur der Daten zu erkennen und trägt zur Erklärbarkeit der Ergebnisse bei. Diese Art der Darstellung ermöglicht die Komplexität der Daten in verständlicher Form darzustellen. Die unterschiedliche Farbgebung ermöglicht Bereiche oder Datenpunkte hervorzuheben, um wichtige Informationen besser zu erkennen. Die Intensität der Farbgebung ermöglicht bei einer 3D-Darstellung zudem einen zusätzlichen Datenwert anzuzeigen. Liniendiagramme eignen sich zur Darstellung zeitlicher oder kontinuierlicher Datenverläufe. Die unterschiedliche Farbgebung ermöglicht eine klare Unterscheidung verschiedener

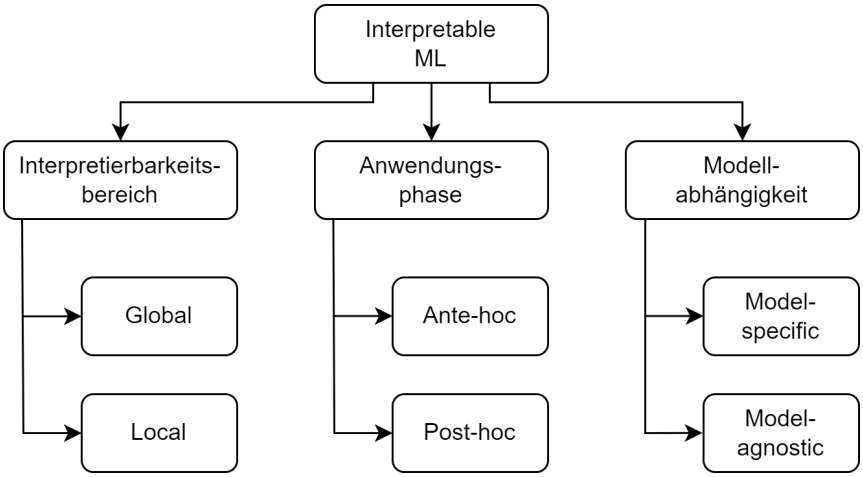


Abbildung 4.1: Die Taxonomie des interpretierbaren maschinellen Lernens umfasst übergeordnet die Kategorien des Interpretierbarkeitsbereichs, der Anwendungsphase und der Modellabhängigkeit mit den jeweiligen Unterteilungen (Eigene Darstellung, in Anlehnung an: [79])

Datenkategorien, wodurch Unterschiede in den Daten einfach erkennbar sind. Die Darstellung von Daten mit Hilfe von Balkendiagrammen hingegen ermöglicht die Visualisierung von Kategorien. Die Verwendung verschiedener Farben dient zur Unterscheidung der Kategorien. Boxdiagramme erlauben die Darstellung von Ausreißern sowie die Visualisierung der Verteilung der Daten innerhalb eines Bereichs. Es existieren verschiedene Konzepte zur Erklärbarkeit von Modellen, die in der Abbildung 4.1 dargestellt und nach folgenden Kriterien unterteilt werden können.

Globale vs. Lokale vs. Kontrafaktische Erklärbarkeit: Im Rahmen der *globalen Erklärbarkeit* wird das Modellverhalten ganzheitlich erklärt. Dabei wird das Verhalten im Allgemeinen auf einer übergeordneten Ebene betrachtet und versucht, dieses zu verstehen. Dies ermöglicht die Erklärung der Auswirkungen der Eingabeparameter auf die maschinellen Lernalgorithmen. Im Gegensatz dazu beschreibt die *lokale Erklärbarkeit* die Identifikation der Merkmale, die zur Vorhersage beitragen. Der hier vorgestellte Erklärungsansatz erlaubt durch die Betrachtung von Datenproben Rückschlüsse auf das Verhalten des Modells, wie in [80] beschrieben. Ein weiterer Ansatz zur Vorhersage ist die *kontrafaktische Erklärbarkeit*. Um eine Anomalie zu erklären, werden hierfür die Gegenfaktoren herangezogen. Im Rahmen dessen wird eruiert, welche Modifikationen des Eingangsmerkmals erforderlich sind, um eine divergierende Entscheidung des Modells zu induzieren, während die übrigen Merkmale konstant gehalten werden Abbildung 4.2).

Ante- vs. Post-hoc-Verfahren: Das *Ante-hoc-Verfahren*, auch *transparent* oder *intrinsisch* genannt, zeichnet sich dadurch aus, dass das Modell aufgrund seiner Struktur interpretierbar ist und als Glasbox-Modell bezeichnet wird. Die

Entscheidungen, die das Modell trifft, basieren auf Parametern und Regeln, die während des Trainings erlernt wurden. Die aus dem Modell abgeleiteten Regeln sind selbsterklärend und nachvollziehbar. Zu diesen Modellen gehören beispielsweise Entscheidungsbäume, lineare Modelle oder regelbasierte Systeme. Wird die Analysemethode erst nach dem Training angewendet, spricht man von einem *Post-hoc-Verfahren*. Dieser Ansatz wird bei komplexen und schwer zu interpretierenden Black-Box-Modellen verwendet. Dabei wird versucht, die Entscheidungen bzw. das Verhalten des Modells als Ganzes zu interpretieren. Das Ziel besteht folglich in der Untersuchung der Abhängigkeiten zwischen den Eingabemerkmalen und den Vorhersagen. Zu den bekanntesten Verfahren dieser Kategorie zählt SHAP [81].

Modell-spezifisch vs. Modell-agnostisch: Eine weitere Unterscheidung kann anhand der Modellabhängigkeit getroffen werden. Diese bezieht sich darauf, ob eine interpretierbare maschinelle Lerntechnik auf jedes Modell oder nur *modellspezifisch* angewendet werden kann. Die Anwendung auf ein spezifisches Modell hat den Vorteil, dass bestimmte Eigenschaften und eine tiefere Interpretierbarkeit ermöglicht werden. Die Erklärungstechnik kann jedoch nur auf einen bestimmten Typ oder ein bestimmtes Modell angewendet werden. Die Anwendung der *modellunabhängigen* Technik (Model-agnostic) kann flexibel auf jedes maschinelle Lernmodell angewendet werden, ohne dass spezifisches Wissen über dessen interne Struktur erforderlich ist. Bei dieser Technik wird das Modell als Black-Box betrachtet, was die Möglichkeit bietet, Modelle mit der gleichen Technik zu erklären [82].

Im weiteren Verlauf dieser Arbeit erfolgt die lokale und globale Erklärbarkeit der Ergebnisse von Machine-Learning-Algorithmen mithilfe des Whitebox-Modells „Explainable Boosting Machine“ (EBM). Zum Vergleich hierzu erfolgt mit dem modellunabhängigen Erklärungsverfahren SHAP (SHapley Additive exPlanations) die Erklärung der Ergebnisse des Modells. SHAP stellt ein weitverbreitetes Verfahren dar, das sowohl auf White- als auch auf Blackbox-Modelle angewendet werden kann.

4.1 Explainable Boosting Machine

Das Explainable Boosting Machine (EBM) Modell ist ein Bestandteil des Frameworks InterpretML und stellt eine Technik des maschinellen Lernens dar. Es wurde entwickelt, um eine hohe Genauigkeit der Erklärbarkeit und ein erklärbares Modell bereitzustellen. Die Basis dieses Algorithmus stellt das verallgemeinertes lineare Modell (Generalized Additive Model – GAM) dar, welches durch die Anwendung von unabhängigen und interpretierbaren Funktionen erklärbar ist. Die Abbildung 4.3 veranschaulicht die Einordnung des Explainable Boosting Algorithmus, der eine vergleichbare Genauigkeit wie der Boosted-Trees Algorithmus aufweist und zudem durch eine hohe Verständlichkeit des Entscheidungsbaums gekennzeichnet ist.

Das verallgemeinertes lineare Modell, dargestellt durch die Gleichung 4.1, ist der Logistic Regression sehr ähnlich und stellt die mathematische Grundlage für EBM dar.

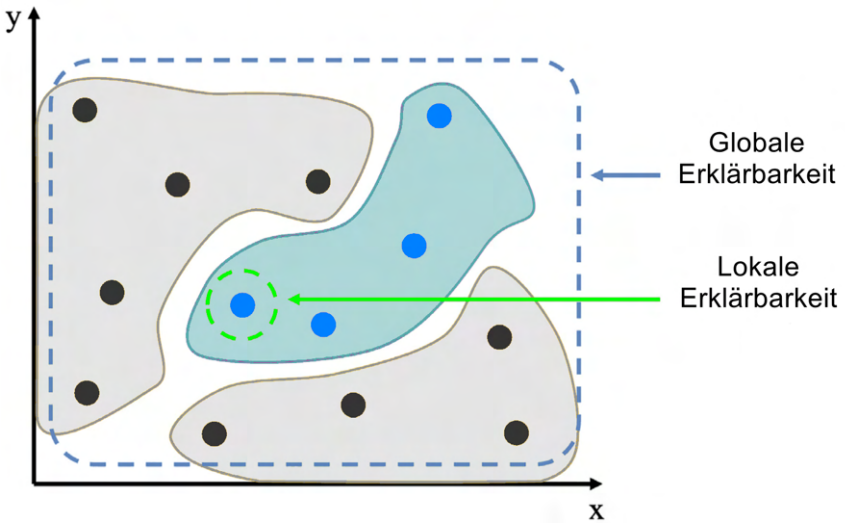


Abbildung 4.2: Die Abbildung veranschaulicht die lokalen und globalen Erklärbarkeit einer Klassifizierungsaufgabe. Die globale Erklärbarkeit berücksichtigt das Modell in seiner Gesamtheit. Demgegenüber fokussiert sich die lokale Erklärbarkeit auf einen einzelnen Datenpunkt (Eigene Darstellung, in Anlehnung an: [79])

$$g(E[y]) = \beta_0 + \sum f_j(x_j) \tag{4.1}$$

Das GAM-Modell wird durch die Verknüpfungsfunktion g an verschiedene Modelle, wie Regression oder Klassifikation, angepasst. Die Funktion wird durch f_j für das Merkmal x_j repräsentiert, während $E[y]$ den erwarteten Wert und β_0 den Bias-Parameter darstellt. GA2Ms stellen die Erweiterung des GAM-Modells mit einem Interaktionsterm dar. Die Darstellung erfolgt durch die Elemente (x_i, x_j) entsprechend Gleichung 4.2 ([84]).

$$g(E[y]) = \beta_0 + \sum f_j(x_j) + \sum f_{ij}(x_i, x_j) \tag{4.2}$$

Die genaue Beziehung zwischen den Eingangsmerkmalen x_j und dem Ziel y kann somit als zweidimensionaler Plot dargestellt werden [85]. EBMs sind baumbasierte, zyklische GA2Ms, welche die Funktionen $f_i(x_i)$ und $f_{ij}(xi, xj)$ der Gleichung GAM02 durch gradient-boosted Ensembles von Bagged Trees ersetzen. Ein Haupteffekt kennzeichnet die direkte Auswirkung eines Eingangsmerkmals auf eine Zielvariable. Hierbei wird ein Baum nur auf ein einziges Merkmal trainiert. Ein Iterationseffekt hingegen stellt die kombinierte Auswirkung auf zwei oder mehrere Merkmale dar. Diese werden auf zwei Merkmale trainiert [86]. Zudem kann die Explainable Boosting Maschine automatisch die Interaktionsterme (x_i, x_j) erkennen und einbeziehen. Dies dient zur Erhöhung der Genauigkeit und zum Erhalt der Verständlichkeit. Die

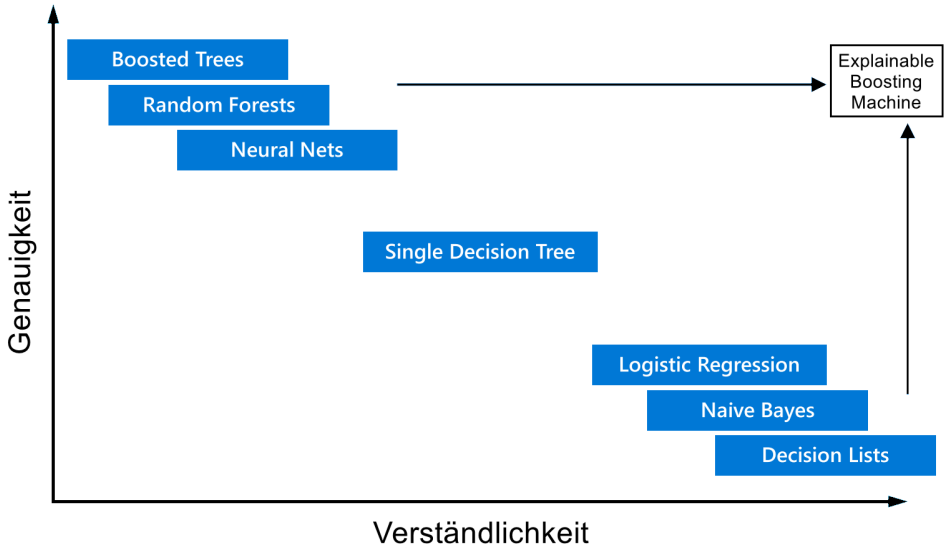


Abbildung 4.3: Die Explainable Boosting Machine ermöglicht eine hohe Genauigkeit, die mit der von Random Forest vergleichbar ist. Zudem bietet sie die Verständlichkeit einer regelbasierten Methode (Eigene Darstellung, in Anlehnung an: [83])

Verständlichkeit ist in diesem Kontext als die Eigenschaft definiert, dass durch die Berücksichtigung der Interaktionsterme die Gesamtinterpretierbarkeit und Modelltransparenz erhalten bleibt. Beim der Explainable Boosting Maschine erfolgt das Trainieren der Bagged Trees mit den Merkmalen in einer geringer priorisierten Reihenfolge im Round-Robin-Verfahren (RR). Dieses gewährleistet, dass die Reihenfolge keinen Einfluss auf das Ergebnis hat. Wie nach dem Grandiet-Boosting-Verfahren werden viele Bäume nacheinander trainiert und der Fehler des vorangegangenen Baums erklärt. Hierbei trainiert jeder Baum jeweils nur ein Merkmal mit einer geringen Priorität. Die Abbildung 4.4 stellt den prinzipiellen Ablauf des Algorithmus dar, der im nachfolgenden Abschnitt beschrieben wird.

In der ersten Iteration wird ein kleiner Baum mit dem Merkmal 1 ($feature_1$) erstellt und trainiert. Dieser Baum enthält lediglich dieses Merkmal. Der Residual, in der Abbildung 4.4 als „res“ bezeichnet, ist die Differenz zwischen den tatsächlichen Wert der Zielvariable und den vom Modell vorgesagten Wert. Im nächsten Schritt wird die Vorhersage x_i dem Baum des Merkmals 2 ($feature_2$) übergeben, um die Beitragsfunktion $f_i(x_i)$ zu aktualisieren. Dieser Schritt wird bis zum letzten Merkmal wiederholt und schließt mit einer Iteration ab. Das Ziel ist, mit jeder Iteration den Wert der Residuen zu minimieren. Die Durchführung von bis zu 10.000 Iterationen ist dabei ohne signifikante Performanceprobleme möglich. Die Bäume wurden jeweils nur mit einem Merkmal trainiert und anschließend zu einem Graphen pro Merkmal zusammengeführt. Das Resultat ist eine Reihe von Graphen, welche das Modell dar-

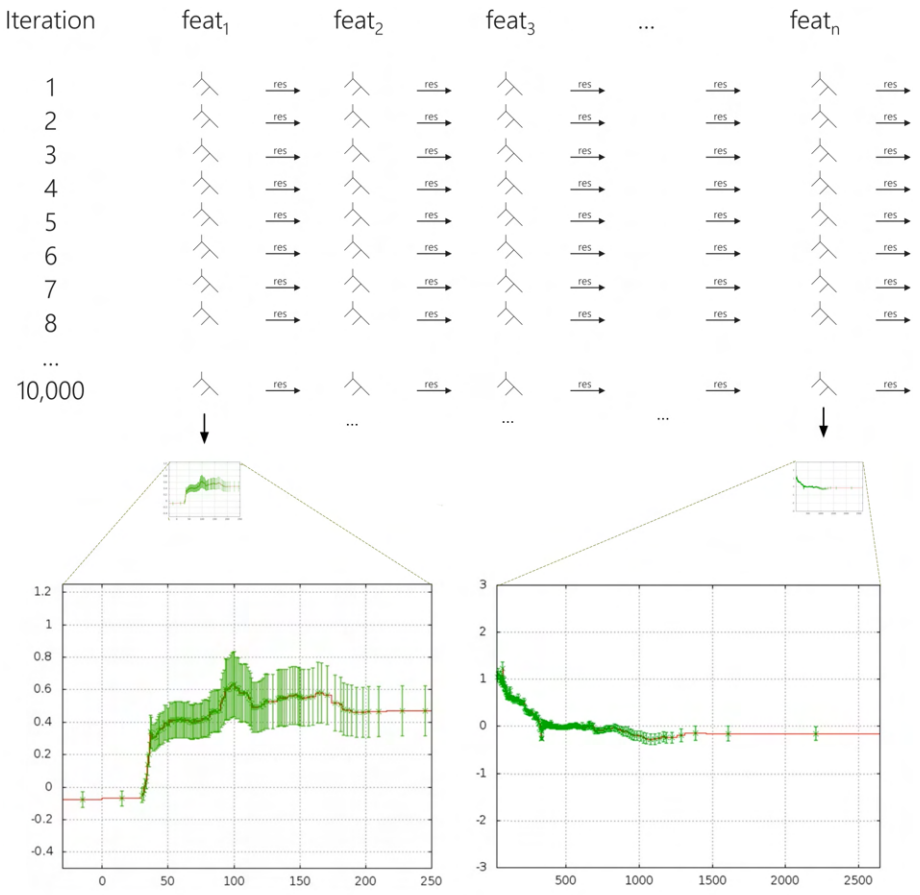


Abbildung 4.4: Prinzipielle Darstellung der Explainable Boosting Machine mit den einzelnen Iterationsschritte 1 bis 10.000. Die Graphen zeigen die zusammengefassten Bäume der Merkmale, die exemplarisch dargestellt sind (Eigene Darstellung, in Anlehnung an: [83]).

stellen. Die Betrachtung der Interaktionen von Merkmalen zueinander erfolgt durch Merkmals-Paare. Dies ermöglicht eine tiefere Analyse der Beziehungen zueinander und führt zu einem verbesserten Verständnis der Modellvorhersagen. Zur Bestimmung der Wechselwirkung der Merkmale zueinander wird der FAST-Algorithmus (Features from Accelerated Segment Test) verwendet, um die Paare auf der Grundlage ihrer Beziehung zu den Restwerten zuzuordnen [83]. Ausgewählt werden hierbei jeweils die besten Paare. Der Ablauf entspricht dem in der Abbildung 4.4 beschriebenen GA2Ms-Algorithmus. Der wesentliche Unterschied besteht darin, dass jeder Baum auf zwei Merkmale trainiert wird **??**. Für das Merkmal $feat_1$ wird ein Merkmalspaar $Pair_n(f_a, f_b)$ usw. verwendet.

4.2 Shapley-Wert

Der Shapley-Wert stammt aus der kooperativen Spieletheorie und wird zur Berechnung des Gewinns verwendet, um diesen zwischen Spielern am fairsten aufzuteilen. Der wesentliche Vorteil des Shapley-Werts besteht in der fundierten mathematischen Grundlage des Mechanismus.

Die Berechnung der Shapley-Wert es kann durch folgendes Beispiel mit den drei Personen Bernd, Chris und Doris erläutert werden. Es wird davon ausgegangen, dass alle Personen zum gleichen Zeitpunkt einer Koalition beitreten und die Fähigkeiten der Personen sich nicht gegenseitig beeinflussen.

Beispiel:

Die drei Personen Bernd (B), Chris (C) und Doris (D) erwirtschaften für sich allein jeweils einen Gewinn von 120 Euro. Jeder dieser Personen hat unterschiedliche Fähigkeiten, sodass sie, wenn sie zusammenarbeiten mehr erwirtschaften können. Die Zweierkoalition zwischen Bernd und Chris ermöglicht einen Gewinn von 170 Euro und zwischen Chris und Doris beträgt dieser 160 Euro. Der Gewinn bei einer Konstellation von Bernd und Doris beträgt 180 Euro. Bei einer Dreikoalition erhöht sich der Gesamtgewinn auf 280 Euro. Die möglichen Kombinationen dieser Koalitionen ergibt sich aus den Permutation P, die durch $P(n) = n!$ bestimmt werden. n stellt die Anzahl der Personen dar. So ergeben sich unter der Berücksichtigung der Reihenfolge sechs Kombinationen: BCD, BDC, CBD, CDB, DBC, DCB (in Anlehnung an: [87]).

Der Shapley-Wert wird berechnet, indem der Durchschnitt der Differenzen aus allen Kombinationen gebildet wird ([88]). Dies ermöglicht eine faire Verteilung des Gewinns auf die Teilnehmer in Abhängigkeit von deren Fähigkeiten. So können die Shapley-Werte auf das Ausgangsproblem eines Modells übertragen werden, indem die Teilnehmer auf die Merkmalsausprägungen und das Ergebnis der Koalition auf die Schätzungen des Modells bezogen werden. Hierdurch kann die Wichtigkeit eines Merkmals auf die Vorhersagewerte eines Modells mittels Shapley-Wert ermittelt werden. Der Shapley-Wert für einen Spieler kann wie folgt berechnet werden ([78]):

$$\phi_j = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|!(N - |S| - 1)!}{N!} (v(S \cup \{j\}) - v(S)), \quad j = 1, ..., N \tag{4.3}$$

$\phi_j(v)$ bezeichnet den Shapley-Wert und somit den Anteil, den jedes Teammitglied j erhält. $S \subseteq N = \{1, ..., N\}$ ist die Teilmenge, die aus $|S|$ Teammitgliedern besteht. N bezeichnet die Koalition der Teammitglieder. $v(S)$ ist der Wert der Teilmenge von

S. $\upsilon : P(N) \mapsto \mathbb{R}$ stellt die Abbildung alle möglichen Koalitionen von N Spielern auf eine reelle Zahl dar ([89]).

Die Grundlage der erörterten Gleichung 4.3 stellen die vier Axiome dar, die von Lloyd Shepley ([90]) aus der Betrachtung einer gerechten Verteilung abgeleitet wurden. Zudem bewies Shapley, dass die Formel, die auf der Basis der Axiome beruht, eine eindeutige Lösung liefert. Diese vier Axiome, die nachfolgend erläutert werden sind Optimalität, Symmetrie, Linearität und Null-Spieler. Diese stellen im weiteren die Definition von Fairness in Bezug auf Auszahlung des Gewinns dar.

- **Optimalität**: Das Axiom sagt aus, dass die Summe der einzelnen Auszahlung an die Teilnehmer gleich der Gesamtauszahlung sein muss.

  $$\sum_{j \in N} \phi_j = \upsilon(N)$$

- **Symmetrie**: Diese besagt, dass wenn zwei Teilnehmern i und j die gleiche Auszahlung erhalten, diese identisch sind. Identisch bedeutet in diesen Zusammenhang, das die Grenzbeträge der Teilnehmer gleich sind.

  Wenn $\upsilon(S \cup \{i\}) = \upsilon(S \cup \{j\}$ für alle $S \subseteq N$, dann gilt $\phi_i(\upsilon) = \phi_j(\upsilon)$.

- **Linearität**: Diese beschreibt, dass bei einem Spiel mit den Funktionen υ_1 und υ_2 die Shapley-Werte für die Summe der Spiele als Summe der Shapley-Werte ausgedrückt werden kann.

  $$\phi_{j,\upsilon_1+\upsilon_2} = \phi_{j,\upsilon_1} + \phi_{j,\upsilon_2}$$

- **Null-Spieler**: Besagt, dass für einen Spieler der keinen Beitrag zum Gesamtgewinn der Koalition leistet, der Shapley-Wert Null ist.

  Wenn $\upsilon(S \cup \{j\}) = \upsilon(S)$ für alle $S \subseteq N$, dann ist $\phi_j = 0$

4.3 SHAP

Die Erklärbarkeit komplexer Modelle, wie beispielsweise Ensemble- oder Neuronale Netze, stellt eine besonders hohe Hürde dar. SHAP (SHapley Additive exPlanations) bezeichnet eine Methode zur Post-hoc-Interpretierbarkeit, welche auf Basis der Shapley-Werte Erklärungen generiert. Auf diese Weise lassen sich Einsichten in das Vorhersageverhalten komplexer Modelle gewinnen. Um diese zu erreichen, verwendet SHAP den Shapley-Wert, um jedem Merkmal des Vorhersagemodells einen Wichtigkeitswert zuzuweisen. Die von SHAP bereitgestellten Erklärungen lassen sich sowohl auf lokaler, als auch auf globaler Ebene anwenden. Das SHAP-Python-Paket, welches die SHAP-Wert-Variable bereitstellt, besteht aus drei Attributen: *.values*, *.base_values und .datafields*. Das Attribut *.values* stellt eine $n \times p$ Matrix dar und repräsentiert die SHAP-Werte. *.base_values* stellt die die durchschnittliche Vorhersage für jeden Datenpunkt dar und *.data* repräsentiert die Merkmalswerte [91].

4.4 Additive Feature-Attributionsmethode

Die Additive Feature-Attributionsmethode bezeichnet eine Klasse von Methoden, die den Vorhersagewert aufgrund von Eingangsmerkmalen auf Grundlage der Shapley-Werte berechnen. Ein einfaches Modell ist für sich selbst die beste Erklärung, da es sich selbst perfekt darstellt und zudem leicht zu verstehen ist ([92]). Die perfekte Selbstdarstellung basiert auf der Tatsache, dass bei einem einfachen Modell die Struktur und die Gewichte der Merkmale klar ersichtlich sind. Bei komplexeren Modellen kann jedoch aufgrund der Komplexität nicht das ursprüngliche Modell verwendet werden. Ein Lösungsansatz stellt eine interpretierbare Annäherung durch ein einfacheres Erklärungsmodell dar. Im nachfolgenden wird die lokale Erklärung verwendet, um die Vorhersage f_x auf x abzubilden. Das ursprüngliche Modell wird als f und das Erklärungsmodell als g bezeichnet. Die vereinfachte Eingabe x', die im Regelfall von den Erklärungsmodellen verwendet wird, erfolgt mittels der Abbildung $x = hx(x')$ auf die ursprünglichen Eingaben.

Erklärbare lokale Methoden zielen darauf ab, dass $g(z') \approx f(hx(z'))$ immer gültig ist, wenn $z' \approx x'$.

$$g(z') = \phi_0 \sum_{i=1}^{M} \phi_i z'_i \tag{4.4}$$

Wobei $z' \in \{0,1\}^M$, $\phi_i \in \mathbb{R}$ und M die Anzahl der vereinfachten Merkmale darstellt ([92]). Mit Hilfe des gezeigten Verfahrens lassen sich viele Modelle wie LIME (Local Interpretable Model-agnostic Explanations) beschreiben. Die Klasse der Additive Feature-Attributionsmethode hat Eigenschaften, die dem Shapley-Wert sehr ähnlich sind und mit den genannten Charakteristiken eine eindeutige Lösung darstellen. Hierbei kann eine gleichwertige Darstellung zu den Shapley-Wert hergestellt werden ([92]).

- **Lokale Exaktheit**: Diese kann mit der Effizienz von Shapley gleichgesetzt werden. Es erfordert, dass das Erklärungsmodell $g(x')$ gleich dem Originalmodell $f(x)$ ist. x einspricht hierbei einen bestimmten Eingabewert. Zudem stellt $\phi_0 = f(h_x(0))$ die Modellausgabe dar, wenn alle vereinfachten Eingaben fehlen.

  $$f(x) = g(x') = \phi_0 \sum_{i=1}^{M} \phi_i x'_i$$

- **Ungenauigkeit:** Wird mit dem Null-Spieler gleichgesetzt und bedeuten, dass einem fehlenden Merkmal keine Auswirkung zugeschrieben wird.

  $$x'_i = 0 \Rightarrow \phi_i = 0$$

- **Konsistenz:** Die besagte Vorschrift besagt, dass sich die Zurechnung der Eingangsmerkmale nicht verringern darf, sofern durch die Änderung des Modells

der Betrag eines vereinfachten Eingangsmerkmals gegenüber den anderen Eingangsmerkmalen gleich bleibt oder steigt.

$Für f_x(z') = f(h_x(z'))$ und $z' \setminus i$ gilt $z'_i = 0$.

Für zwei Modelle f und f' gilt, dass wenn

$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i)$

für alle Eingangsmerkmale $z' \in \{0,1\}^M$, dann gilt $\phi_i(f', x) \geq \phi_i(f, x)$.

Die nachfolgende Gleichung 4.5 beschreibt das Modell, welches die drei Eigenschaften Lokale Exaktheit, Ungenauigkeit und Konsistenz erfüllt ([93]) und somit gleich den Shapley-Wert ist.

$$\phi_i(f, x) = \sum_{z' \in x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z' \setminus i)] \tag{4.5}$$

Die Gleichung 4.5 stellt die Grundlage für Additive Feature-Attributionsmethoden für SHAP dar. Neben Shap und Shapley-Werten existieren noch weitere Methoden, wie beispielsweise die Permutation Feature Importance (PFI) oder der Partial Dependence Plots (PDPs), die zur globalen Erklärbarkeit von Maschine-Learning Algorithmen herangezogen werden können. Diese werden in dieser Arbeit jedoch nicht gesondert berücksichtigt. Anstatt dessen wird Shapley-basierte Feature Importance verwendet, die mit Hilfe der Aggregation von Shapley-Werten die globale Erklärbarkeit von Maschine-Learning–Algorithmen ermöglichen [94].

5 Versuchsumgebung

Im Rahmen einer Studie wurden beim BSI Kanalzustandsinformation in einer Versuchsumgebung aufgezeichnet, die für die vorliegende Arbeit zur Verfügung gestellt werden. Die einheitliche Datenstruktur gewährleistet die Reproduzierbarkeit der Ergebnisse. Im Folgenden werden die Versuchsumgebung sowie die verwendeten Komponenten beschrieben.

5.1 Umgebung

Um möglichst realistische Kanalzustandsinformationen zu erhalten, wurden Bewegungsabläufe in einem Besprechungsraum des BSI erfasst. In einem Raum (Abbildung 5.1) mit einer durchschnittlichen Breite von ca. 8,5m und einer Tiefe von ca. 5m wurden die Komponenten aufgebaut. Die verwendete Hardware besteht aus einem ESP32 (AP), der als Access-Point arbeitet, zwei ESP32-Stationen (STA1/STA2) und einem Analyse-Laptop. Der Laptop ist mit der Software zur Aufzeichnung der CSI-Daten ausgestattet und befindet sich außerhalb des Besprechungsraumes, um Interferenzen zu vermeiden. Der Accesspoint wurde zu diesem Zweck mit einer Höhe von 70 cm auf der linken Seite des Raumes positioniert. Auf der gegenüberliegenden Seite befinden sich die beiden Stationen STA1 und STA2 mit einer Höhe von jeweils 64 cm. Die Positionen, von denen aus die Bewegungen durchgeführt wurden, sind mit P1 – P3, die im weiteren Verlauf der Arbeit durch #1 – #3 bezeichnet werden, definiert.

Die für den Versuchsaufbau verwendeten Microcontroller ESP-32 wurden von der Firma ESPRESSIF entwickelt. Das kostengünstige und leistungsfähige Gerät verfügt über WLAN und bietet zusätzlich die Möglichkeit die Kanalzustandsinformationen auszugeben. Die notwendige Konfiguration, wie die Konfiguration des WLAN-Netzwerks und die Aktivierung der CSI-Ausgabe wird mit dem Espressif IoT Development Framework durchgeführt. Die erforderlichen Einstellungen sind in der Literatur unter [95] ausführlich beschrieben. Nach dem Start der Stationen STA1 und STA2 verbinden sich diese mit dem vom Accesspoint aufgebauten WLAN-Netzwerk. Die SID des WLAN-Netzwerks und WLAN-Key wurden zuvor mit der Konfiguration der Stationen programmiert und so konfiguriert, dass UDP-Pakete an den Accesspoint versendet werden. Auf diese Weise ist der Accesspoint in der Lage, die CSI-Informationen der Übertragung zu ermitteln und an den Laptop außerhalb des Raums zu übertragen. In der Abbildung 5.2 ist der Aufbau der Versuchsumgebung mit den einzelnen Komponenten dargestellt.

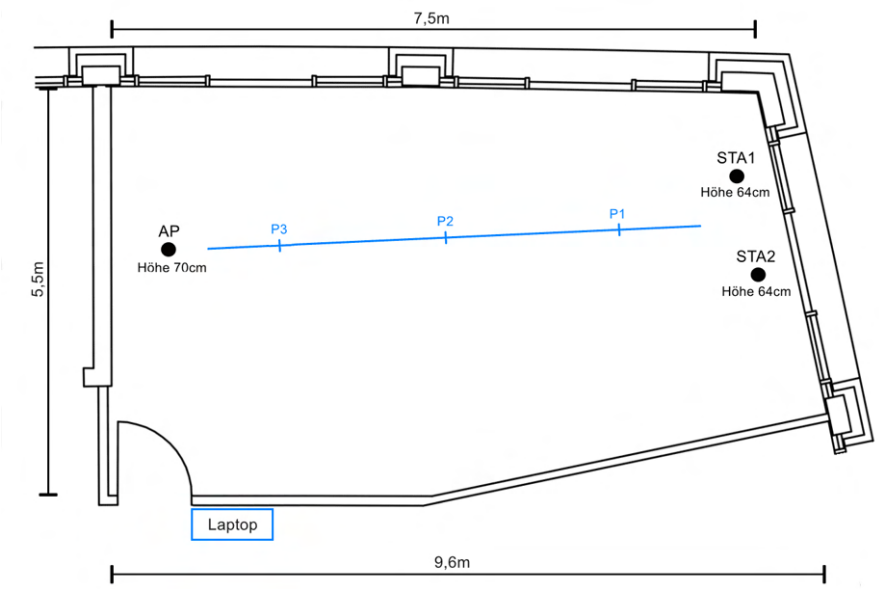


Abbildung 5.1: Die Abbildung zeigt die Versuchsumgebung mit den verwendeten Komponenten. Links ist der Accesspoint (AP) angeordnet, auf der rechten Seite befinden sich die beiden Stationen (STA1 und STA2). Die Buchstaben P1 bis P3 bezeichnen die Positionen, an denen Bewegungen ausgeführt wurden. Links unten außerhalb der Versuchsumgebung ist das Analyse-Laptop zu sehen.

5.2 Hardware

Die Ausführung der Machine-Learning-Algorithmen erfolgte auf einem leistungsfähigen XMG-Notebook (Tabelle 5.1). Die technischen Daten des Gerätes sind in der Tabelle 5.1 aufgeführt. Auf die daraus resultierende Performance der Algorithmen wird im weiteren Verlauf der Arbeit noch näher eingegangen.

Tabelle 5.1: Die technische Daten des verwendeten XMG-Notebook.

Hersteller	XMG
Type	XMG-NEO-E23
Anzahl der CPUs	1
CPU Type	13th Gen Intel(R) Core(TM) i9-13900HX / bis zu 5,4 GHz
Anzahl Cores	32
Arbeitspeicher	64 GB
Grafikkarte	NVIDIA GeForce RTX

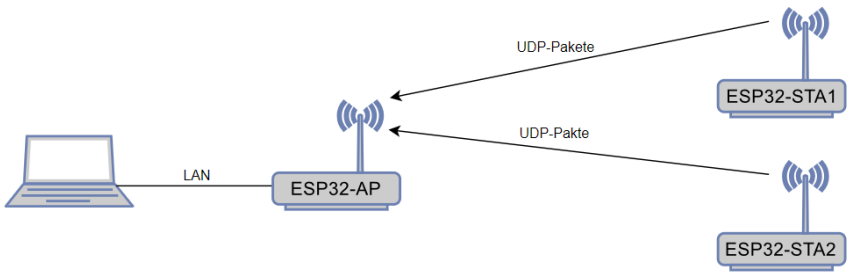


Abbildung 5.2: Die Abbildung zeigt die in der Versuchsumgebung verwendeten Komponenten. Links ist das Analyse-Laptop mit einer LAN-Verbindung zum Accesspoint dargestellt. Rechts sind die Stationen 1 und 2 mit der Funkverbindung zum Accesspoint zu sehen.

5.3 Bewegungsabläufe

Die vom BSI bereitgestellten Datensets mit den Kanalzustandsinformationen sind nach verschiedenen Kriterien gegliedert. Dabei werden Bewegungsabläufe von Personen unterschieden, die sich in Bezug auf Bewegungen, Geschwindigkeiten und Richtungen unterscheiden. Es werden zwei Personengruppen unterschieden. Gruppe 1 mit drei Personen führte pro Person 18 verschiedene Testszenarien durch, während die Gruppe 2, die aus fünf Personen besteht, drei Szenarien durchführte. Die Gruppe 1 führte die Bewegungen Gehen, Stehen und Hampelmann durch. Die Geschwindigkeit der Bewegung „Gehen" wurde in drei Varianten erfasst: langsam, normal und schnell. Die übrigen Bewegungen wurden hinsichtlich ihrer Richtung und Position unterschieden. Die Richtung wurde in Bezug auf die Position der Testperson zum Accesspoint bzw. zur Station 1 und 2 definiert. Die Sichtlinie zum Accesspoint bzw. Station 1 und 2 wird als LLOS (Length Line of Side) bezeichnet, während die um 90 Grad gedrehte Richtung als QLOS (Quer Line of Side) bezeichnet wird. Bei der Gruppe 2 gibt es, wie bei der Gruppe 1, je Person drei verschiedene Testszenarien des Gehens. Die Bewegung erfolgte hierbei ebenfalls in den unterschiedlichen Geschwindigkeiten langsam, normal und schnell. Zur besseren Übersichtlichkeit wurden diese in Form einer Mindmap dargestellt (siehe Abbildung 8.1).

6 Bewegungserkennung mittels Maschine-Learning Methoden

6.1 Datenset

Das vom BSI zur Verfügung gestellte Datenpaket umfasst 70 verschiedene Datensets mit einer Aufzeichnungsdauer von jeweils ca. 60 Sekunden. Jedes einzelne Datenset beschreibt eine Einzelbewegung wie Gehen, Stehen usw. Alle Datensets weisen eine einheitliche Struktur auf und enthalten 166 Datenfelder sowie pro aufgezeichneter Einzelbewegung im Durchschnitt ca. 48.700 Samples. Die Abbildung 6.1 zeigt den Ausschnitt der Rohdaten einer einzelnen Bewegung. Die Datenfelder der Kanalzustandsinformation sind in den Tabellen 8.2 und 8.3 detailliert beschrieben. Die Datenfiles wurden von 1 bis 70 durchnummeriert, sodass eine Zuordnung des Labels zum Bewegungsszenario und zum Datenfile möglich ist. Eine Zuordnung von Dateinamen und Szenarien erfolgt über Pandas-Datenframes, die als CSV-Dateien gespeichert werden. Dadurch wird eine flexible Anpassung der Szenarien ermöglicht und die Verwendung in andere Skripte vereinfacht.

6.2 Datenbereinigung

Die Erstellung der Skripte erfolgt in der Webanwendung Jupyter Notebook. Die Anwendung bietet unter anderem die Möglichkeit, Python-Skripte und Maschine-Learing-Algorithmen von scikit-learn einzubinden und auszuführen. Aufgrund der Gesamtgröße der Datenfiles (2,68 GB) und der Möglichkeit, die Datensets flexibel zu kombinieren, erfolgte die Datenbereinigung schrittweise. Zunächst wurden Samples mit unbekannten MAC-Adressen (Media-Access-Control-Adresse) aus den Datensets entfernt. Diese wurden während der Datenaufnahme empfangen und sollten bei der Analyse der Datensets nicht berücksichtigt werden. Eine MAC-Adresse stellt eine eindeutige Kennung dar, welche ein Netzwerk- interface eindeutig identifiziert. Wie in der Abbildung 6.1 ersichtlich ist, enthalten die Rohdaten Merkmale, die für die weitere Verwendung nicht erforderlich sind. Dazu gehören Nullträger, die Angabe der CSI-Datenlänge usw. Um die Berechnung der Amplitude zu ermöglichen, wurden unerwünschte Zeichen aus den Datensets entfernt. Hierbei wurde zudem die Anzahl der Dezimalstellen auf vier begrenzt. Des Weiteren erfolgte in diesem Schritt das Labeln der Datensets, welche die Grundvoraussetzung für das überwachte Lernen (Unterabschnitt 3.1.1) darstellt.

timestamp_received	source_mac_addr	csi_re	csi_len	csi_subc	csi_su	csi_su	csi_su	csi_su	csi_su	csi_subcarrier_6	csi_subcarrier_7
2023-06-28 14:08:21.991981+00:00	ec:94:cb:6e:73:8c	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(-14, -7)	(-14, -7)
2023-06-28 14:08:21.992582+00:00	ec:94:cb:6e:73:8c	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(2, -17)	(2, -17)
2023-06-28 14:08:21.993904+00:00	ec:94:cb:6e:7c:64	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, -16)	(0, -16)
2023-06-28 14:08:21.995626+00:00	ec:94:cb:6e:73:8c	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(-7, 13)	(-6, 13)
2023-06-28 14:08:21.997945+00:00	ec:94:cb:6e:73:8c	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(-14, 2)	(-15, 2)
2023-06-28 14:08:21.999222+00:00	ec:94:cb:6e:7c:64	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(9, -15)	(8, -15)
2023-06-28 14:08:22.000777+00:00	ec:94:cb:6e:73:8c	83	128	(83, -80)	(4, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(-15, -5)	(-15, -5)

Abbildung 6.1: Diese zeigt einen Auszug aus des CSI-Rohdatensets. Verwendet wurden für die Spalten *source_mac_addr* und die *csi_subcarrier_6* bis *csi_subcarrier_58*. Diese enthalten die Informationen, die im späteren Verlauf bei den Machine-Learning-Algorithmen zum Einsatz kamen.

Die relevanten CSI-Daten in den Datensets bestehen aus komplexen Zahlen (Imaginär- und Realteil) [96] die berücksichtigt werden müssen. Dazu wurden diese Werte aufbereitet und daraus die Amplitudenwerte nach der Gleichung 2.17 berechnet. Des Weiteren wurde eine 3D-Ansicht (Abbildung 6.2) des Datensets „Einer Person im Gehen“ erstellt, welche die unregelmäßige Struktur der Daten darstellt.

6.3 Entfernen von Ausreißern

Nach der Datenbereinigung und Ermittlung der Amplitudenwerte wurden die Datensets mit einer grafischen Untersuchung auf Ausreißer hin untersucht (Unterabschnitt 3.3.1). Hierbei handelt es sich um Werte in den Datensets, die signifikant von den übrigen Daten abweichen. Es zeigt sich durch die grafische Darstellung der Datensets, dass mehrere Ausreißer vorhanden sind. Die Abbildung 6.3 des Datensets „Einer Person im Gehen“ zeigt exemplarisch die Ausreißer. Zur Eliminierung der Ausreißer wurden der Hampel-Filter von scikit-learn verwendet. Der Parameter „Windowssize“ definiert die Größe des gleitenden Fensters der, welches für die Erkennung von Ausreißern verwendet wird.

Zur Kontrolle der korrekten Größe des Fensters für alle Datensets wurden diese nacheinander überprüft. Der Wert der Fenstergröße des Hampel-Filters wurde hierzu von 1 bis 19 schrittweise erhöht und die Maximalwerte für die anschließende Auswertung gespeichert. Die Darstellung der ermittelten Werte erfolgte durch einen Boxplot. Bei einem Wert von 6 für die Fenstergröße zeigten sich keine Ausreißer mehr (Abbildung 6.4).

Die bereinigten Datensets wurden als Szenarien-Files zusammengefasst, um Personenidentifikation, Bewegungsdetektion und positionsabhängige Bewegungsdetektion zu unterscheiden (Tabelle 8.6, 8.7, 8.8). Die zu den Szenarienfiles zusammengefassten Datensets sind in der Tabelle 8.9 aufgeführt. Ein Ausnahme stellt das Szenario 10 dar. Hierbei wurden alle Bewegungen einer Person an jeweils einer Position (#1,#2,#3) zu jeweils einem neuen Datenset zusammengefasst. Anschließend wurden diese in das Szenario10 integriert (Tabelle 8.10).

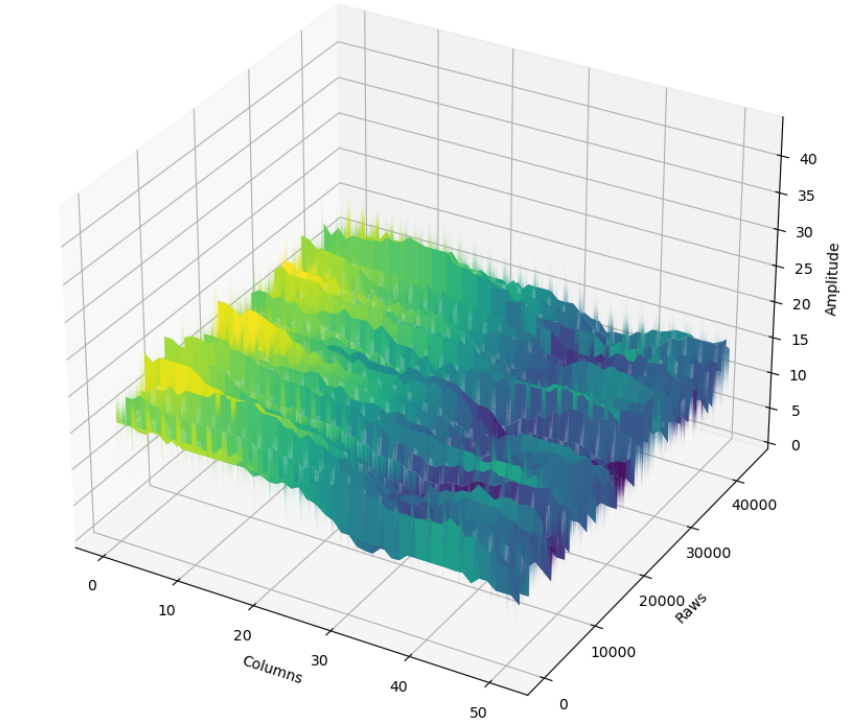


Abbildung 6.2: 3D-Ansicht des Datensets „Person-1 Gehen langsam LLOS“. Die Merkmale werden durch die Angabe der Columns dargestellt. Die Anzahl der Samples beziehen sich auf die Angabe Raws. Nicht unmittelbar sichtbar sind die Ausreißer in den Daten. Der Skalenwert der Amplitude lässt jedoch auf das Vorhandensein von Ausreißern im Datenset schließen.

6.4 Ermittlung der Hyperparameter

Im weiteren Verlauf werden die in Unterabschnitt 3.2.1 beschriebenen Hyperparameter als Grundlage für die ausgewählten Maschine-Learning Algorithmen ermittelt. Hierzu wurde für jedes Szenario das Datenset in 30% Test- und 70% Trainingsdaten aufgeteilt. Die Pipeline von scikit-learn ermöglicht einen mehrstufigen Workflow und verbessert die Performance des Prozesses. Dazu werden der StandardScaler, die Dimensionsreduktion und der Maschine-Learning-Algorithmus in einer Pipeline kombiniert. Im Rahmen der vorliegenden Untersuchung wurde die finale Suche nach den besten Parametern für das Maschine-Learning Modell mit HalvingGridSearchCV durchgeführt. Die ermittelten Hyperparameter wurden in einer CSV-Datei gespeichert um für die weitere Verwendung zur Verfügung zu stehen. Die Tabellen 8.12, 8.13 und 8.15 zeigen die ermittelten Werte der Hyperparameter-Suche.

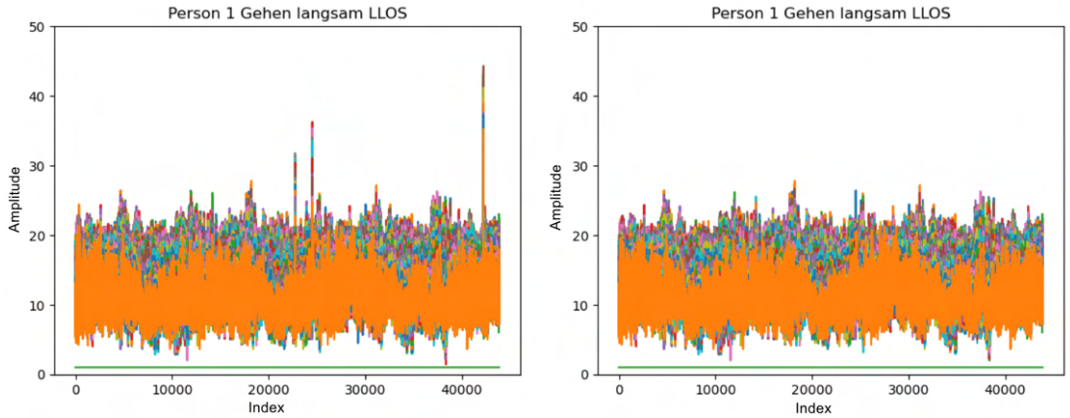


Abbildung 6.3: Die linke Darstellung zeigt CSI-Daten, die Ausreißer beinhalten. Die Y-Achse repräsentiert die Amplitudenwerte der Daten an. Auf der rechten Seite dieser Abbildung werden die CSI-Daten nach der Anwendung des Hampel Filters dargestellt.

6.5 Reduzierung der Dimensionen

Die Reduktion der Datensätze wurde mit Hilfe der Hauptkomponentenanalyse durchgeführt. Die durchgeführten Tests ergaben, dass eine Dimensionsreduktion auf 30 Merkmale eine Vorhersagegenauigkeit von 95,32 % ermöglicht. Der Wert von 95% wurde für die Dimensionsreduktion mit PCA verwendet, da diese auch in anderen Studien [97] verwendet wurde. Die Ergebnisse der Tests sind in der Tabelle 8.11 dargestellt. Die Lineare Diskriminanzanalyse hingegen lieferte in den Tests um ca. 10% schlechtere Ergebnisse als die Principal Component Analysis. Für den Test wurde die Verteilung eines Datensets ermittelt. Es stellte sich heraus, dass die Daten normalverteilt waren. In diesem Fall sollte Lineare Diskriminanzanalyse eigentlich bessere Ergebnisse liefern als die Principal Component Analysis. Eine detaillierte Analyse wurde aus Zeitgründen hierzu jedoch nicht durchgeführt und würde den Umfang der Arbeit sprengen.

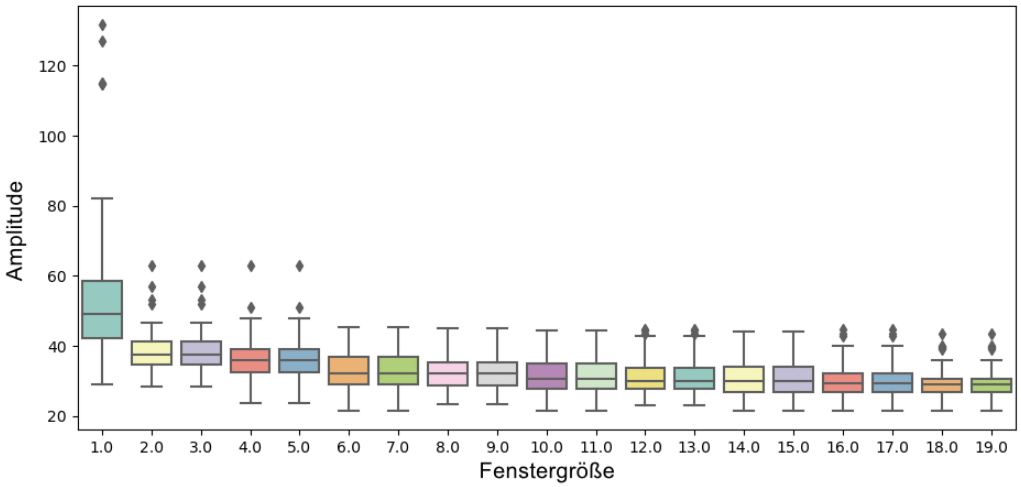


Abbildung 6.4: Die Anwendung des Hampelfilters führt zu dem Ergebnis, dass bei einer Fenstergröße des Hampelfilters von 6,0 keine Ausreißer mehr über den T-förmigen Whisker zu beobachten sind.

7 Resultate

7.1 Datenanalyse und Use-Cases

Im vorliegenden Kapitel erfolgt die Auswertung der Ergebnisse, welche aus zehn verschiedenen Analysen hervorgegangen sind. Diese behandeln Multiklassifikationsprobleme, bei welchen zwischen drei und 18 verschiedenen Klassen unterschieden wird. In einem ersten Schritt wird eine Zusammenfassung der Ergebnisse in tabellarischer Form dargestellt, bevor eine Analyse der Einzelergebnisse erfolgt.

7.1.1 Bewertungsmatrix

Die Ergebnisse der angewendeten Algorithmen wurden in einer Bewertungsmatrix (Tabelle 7.1) dargestellt. Hierzu wurde der F1-Score verwendet und daraus der Durchschnitt über alle F1-Scores der Klassen gebildet. Das zweite Merkmal, das zur Bewertung herangezogen wurde, ist die Laufzeit der Trainings- und Testzeiten, die ein Algorithmus zur Ermittlung des Ergebnisses benötigt. Die ermittelten Ergebnisse sind in der Tabelle 7.1 dargestellt. Grüne Zellen in der Bewertungsmatrix repräsentieren ein positives Ergebnis. Das bedeutet im Falle der Laufzeit eine geringe Anzahl an Sekunden (bis maximal 999 Sec.) und beim F1-Score einen Wert der 0,70 überschreitet. Die Ergebnisse die nicht mehr im grünen Bereich liegen, sind als gelb markiert (Laufzeit von 1000 bis 3600 Sek. bzw. F1-Score von 0,51 bis 0,7). Alle Ergebnisse die eine höhere Laufzeit als 3600 Sekunden (1 Std.) und einen niedrigeren F1-Score als 0,5 haben sind entsprechend rot markiert.

7.1.2 Unterscheidung von Personen

Im Rahmen dieser Auswertung wurde untersucht, inwiefern eine Differenzierung zwischen sieben Einzelpersonen und einer Zweiergruppe möglich ist. Die Einzelpersonen und die Gruppe bewegten sich über einen Zeitraum von etwa 60 Sekunden von Position 1 über Position 2 zu Position 3 und zurück. Der Radius dieser Wegstrecke beträgt ungefähr 6 Meter. Die Bewegungen wurden in Richtung der Sichtlinie (Line of Sight, LLOS) um den Access Point (AP) beziehungsweise die Station (STA) in der Versuchsumgebung (Abbildung 5.1) durchgeführt. Es wird angenommen, dass die Bewegungen der Personen mit den Geschwindigkeiten „langsam, normal und schnell“ jeweils annähernd konstant ausgeführt wurden und somit vergleichbar sind. Auf der Y-Achse der Abbildungen ist jeweils der F1-Score dargestellt, der einen Wertebereich zwischen 0 und 1 umfasst, während die X-Achse die verschiedenen Algorithmen darstellt.

Tabelle 7.1: Die Tabelle zeigt die Bewertungsmatrix für die Szenarien 1 bis 10 mit den entsprechenden farblichen Markierung.

Szenario	Anzahl Samples		Algorithmen: Random Forest	Logistic Regression	K-Nearest Neighbor	Support Vector Machine	Gradient Boosting
1	371150	Laufzeit (Sek.)	124	5	40	12872	1092
		F1-Score	0,52	0,29	0,50	0,51	0,39
2	374083	Laufzeit (Sek.)	132	5	41	13492	1111
		F1-Score	0,50	0,25	0,46	0,48	0,38
3	370736	Laufzeit (Sek.)	132	6	41	13978	1096
		F1-Score	0,46	0,24	0,41	0,46	0,36
4	144197	Laufzeit (Sek.)	43	2	8	1615	153
		F1-Score	0,59	0,48	0,58	0,57	0,53
5	704932	Laufzeit (Sek.)	243	18	138	17324	4224
		F1-Score	0,79	0,40	0,74	0,79	0,68
6	849128	Laufzeit (Sek.)	322	24	198	27799	5520
		F1-Score	0,69	0,33	0,65	0,69	0,57
7	236284	Laufzeit (Sek.)	118	10	82	1519	677
		F1-Score	0,91	0,60	0,85	0,90	0,84
8	238245	Laufzeit (Sek.)	101	7	86	2012	651
		F1-Score	0,88	0,58	0,81	0,87	0,82
9	230405	Laufzeit (Sek.)	98	8	79	1451	646
		F1-Score	0,89	0,62	0,85	0,89	0,83
10	704922	Laufzeit (Sek.)	354	13	135	18042	1286
		F1-Score	0,88	0,61	0,86	0,86	0,79

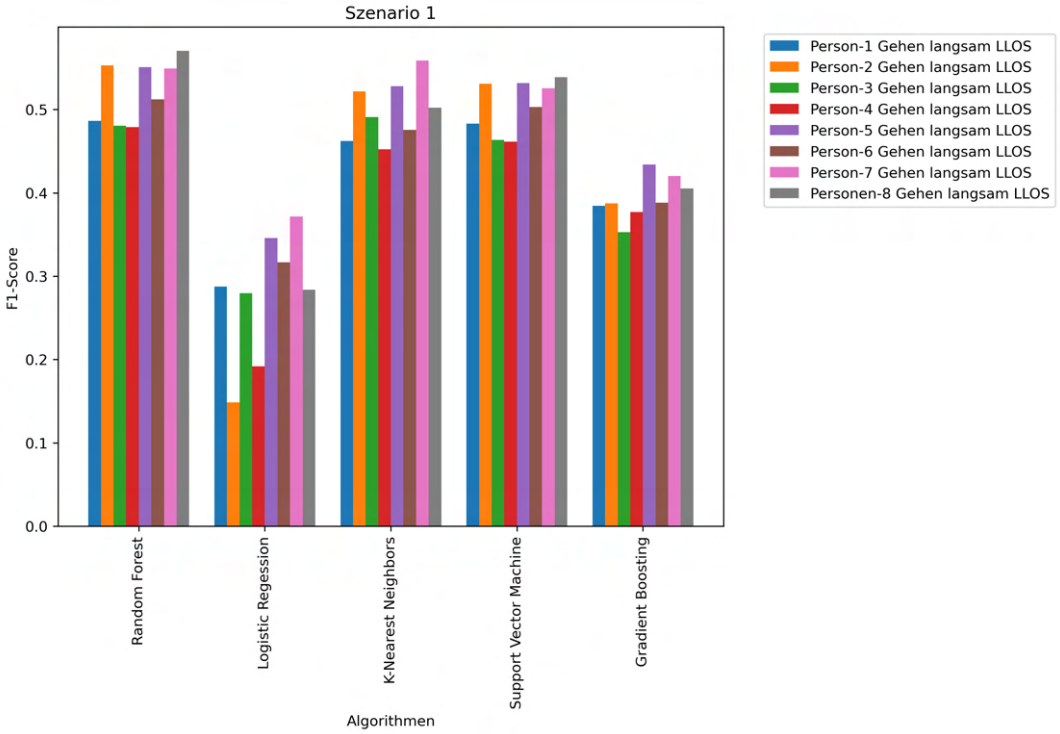


Abbildung 7.1: Ergebnis der Szenario 1 Analyse für die Bewegung „Gehen langsam“ die in Sichtlinie (LLOS) ausgeführt wurde.

Analyse 1: Die Bewegungen „Gehen langsam“ der Personen 1 bis 7 und der Personengruppen 8 wurden in Richtung „LLOS“ ausgeführt. Es ist deutlich zu erkennen, dass der Algorithmus Logistic Regression den niedrigsten durchschnittlichen F1-Score aufweist. Dieser bewegt sich zwischen den Werten 0,15 und 0,37 mit einem Standardabweichung von 0,075. Der F1-Score dieses Machine-Learning-Algorithmus weist auch die größte Spannweite von 0,22 auf. Die besten durchschnittlichen F1-Scores liefern die Algorithmen Random Forest, Support Vector Machine und K-Nearest Neighbor. Die Unterschiede zwischen den durchschnittlichen F1-Scores dieser drei Algorithmen sind minimal.

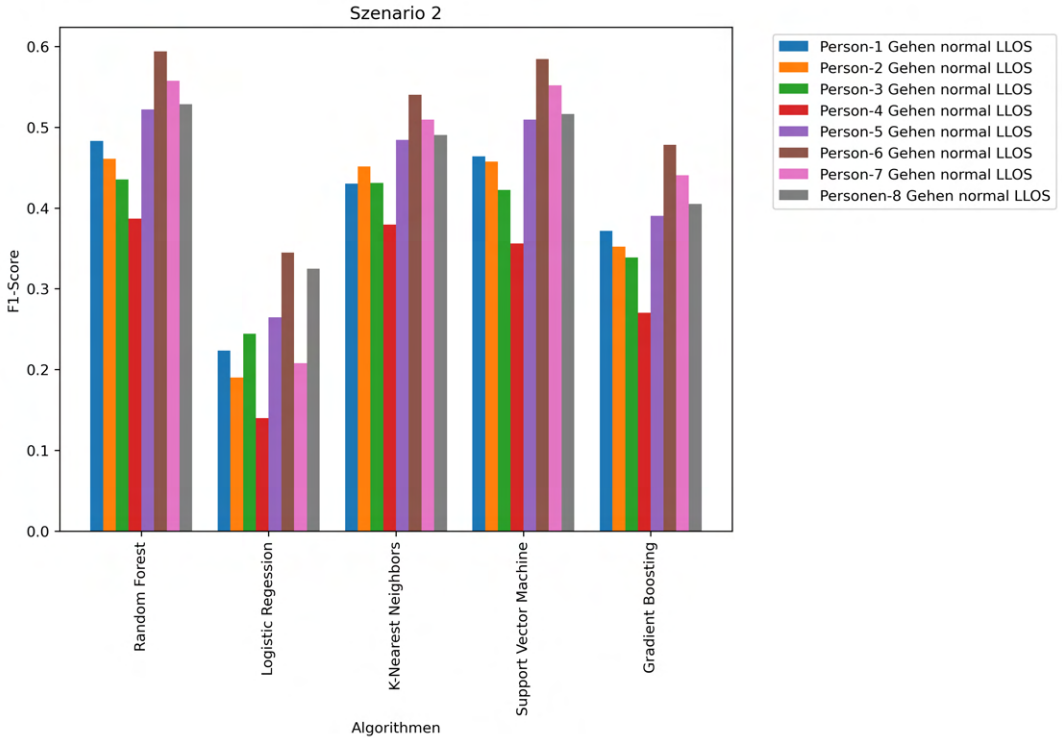


Abbildung 7.2: Ergebnis der Szenario 2 Analyse für die Bewegung „Gehen normal“ die in Sichtlinie (LLOS) ausgeführt wurde.

Analyse 2: Der nächste Analyse ähnelt dem vorhergehenden und unterscheidet sich lediglich in der Geschwindigkeit, mit der die Bewegungen ausgeführt werden. Im Gegensatz zum vorherigen Analyse wurden diese nicht „langsam“, sondern mit „normaler“ Geschwindigkeit durchgeführt. Die Abbildung 7.2 zeigt das Ergebnis der Analyse. Auch in diesem Szenario weist der Random Forest Algorithmus mit einem durchschnittlichen F1-Scores von 0,5 den höchsten Wert mit einer Standardabweichung von 0,07 auf. Die nächstbesten Algorithmen Support Vector Machine und K-Nearest Neighbor erreichen einen durchschnittlichen F1-Score von 0,48 beziehungsweise 0,46. Der niedrigste durchschnittliche F1-Score wird erneut vom Algorithmus Logistic Regression mit einem Wert von 0,24 erzielt. Es ist zudem erkennbar, dass die Person 4 durchgehend bei jedem Machine-Learning-Algorithmus den geringsten F1-Score aufweist.

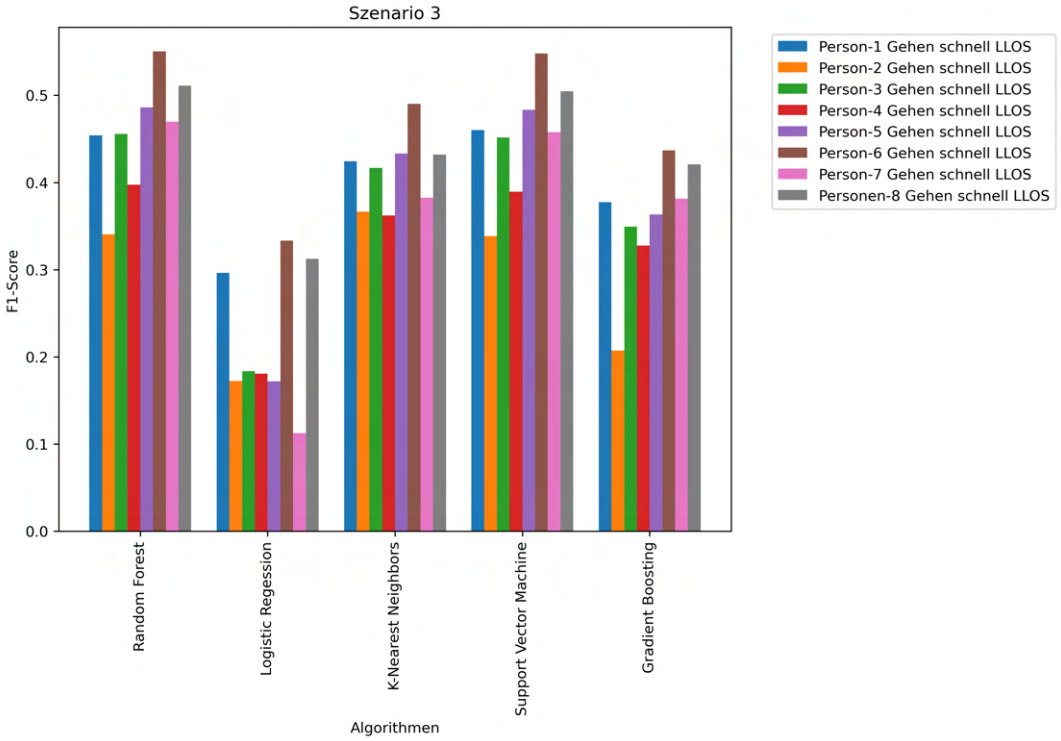


Abbildung 7.3: Ergebnis der Szenario 3 Analyse für die Bewegung „Gehen schnell“ die in Sichtlinie (LLOS) ausgeführt wurde.

Analyse 3:

Die letzte Analyse in dieser Kategorie befasst sich mit dem „Gehen schnell“ der Personen. Der Algorithmus Logistic Regression erzielt erneut den niedrigsten F1-Score von 0,11. Der durchschnittlichen F1-Score beträgt 0,22 mit einer Standardabweichung von 0,08. Die höchsten durchschnittlichen F1-Scores werden von den Algorithmen Random Forest (0,46) und Support Vector Machine (0,45) erreicht.

7.1.3 Unterscheidung von Bewegungen

Innerhalb dieser Kategorie erfolgt eine Differenzierung zwischen Bewegungen, die von einer Person ausgeführt werden.

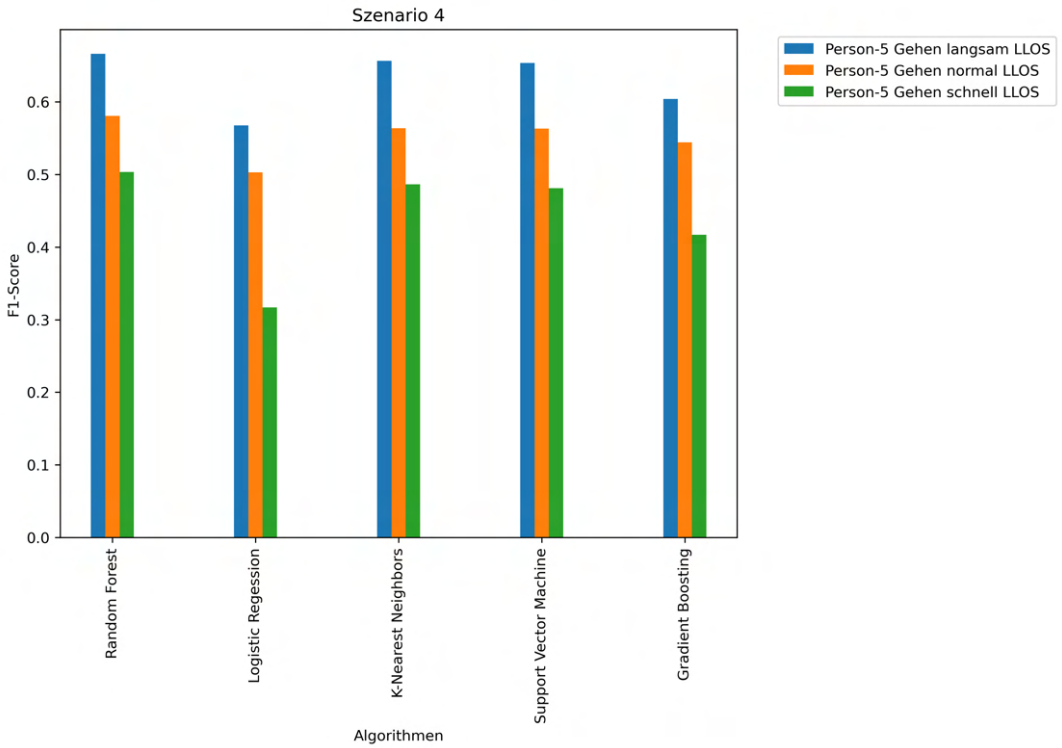


Abbildung 7.4: Ergebnis der Szenario 4 Analyse zur Unterscheidung der Bewegung „Gehen“, die in unterschiedlichen Geschwindigkeiten in Sichtlinie (LLOS) ausgeführt wurde.

Analyse 4: Im Rahmen dieser Untersuchung wurden die Bewegungen der Person 5 analysiert. In diesem Kontext führte die Testperson die Bewegung „Gehen“ in den unterschiedlichen Geschwindigkeiten „langsam, normal und schnell“ aus. Die Ergebnisse dieser Analyse sind in der Abbildung 7.4 dargestellt. Der Algorithmus Random Forest erzielt wiederum den besten F1-Score mit einem durchschnittlichen Wert von 0,58 und einer Standardabweichung von 0,08. Der Unterschied zum Algorithmen K-Nearest Neighbor und Support Vector Machine ist maginal und beträgt beim durchschnittlichen F1-Score 0,01 bzw. 0,02. Es zeigt sich, dass die Unterschiede der Algorithmen im Vergleich zueinander geringer sind, als bei den Szenarien 1 bis 3.

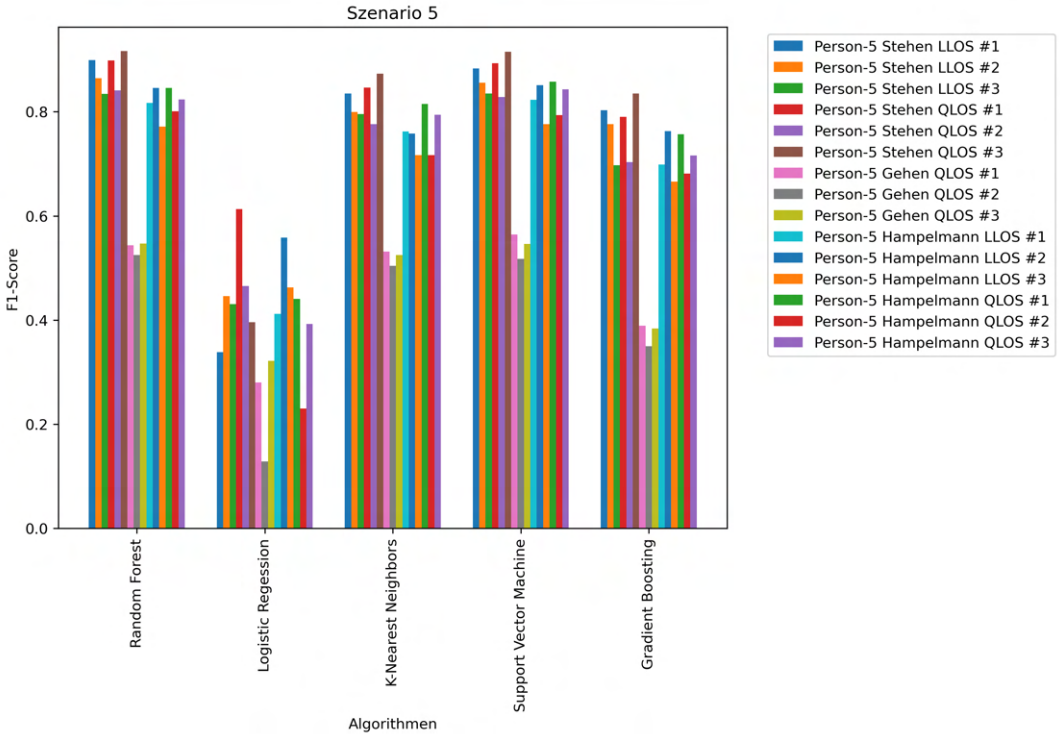


Abbildung 7.5: Ergebnis der Szenario 5 Analyse zur Unterscheidung von Bewegungen von Person 5.

Analyse 5: In Szenario 5 wurden alle Bewegungen, die nicht bereits in Szenario 4 berücksichtigt wurden, einer Bewertung unterzogen. Die durchgeführte Analyse zeigt, dass die beiden Algorithmen Random Forest und Support Vector Machine einen identischen durchschnittlichen F1-Score von 0,79 aufweisen. Des Weiteren ist die Standardabweichung des F1-Scores mit 0,13 gleich. Der Logistische Regressionsalgorithmus zeigt mit einem durchschnittlichen F1-Score von 0,39 und einem Minimalwert von 0,12 die schlechteste Performance. Dieser Trend konnte bei diesem Algorithmus ebenfalls bei den Szenarien 1 bis 3 beobachtet werden. Die höchste Spannweite von 0,49 bzw. 0,48 weisen der Gradient Boosting und der Logistic Regressions Algorithmus auf. Dies ist auf die Ergebnisse der Bewegung „Gehen QLOS #1 #2 #3“ zurückzuführen.

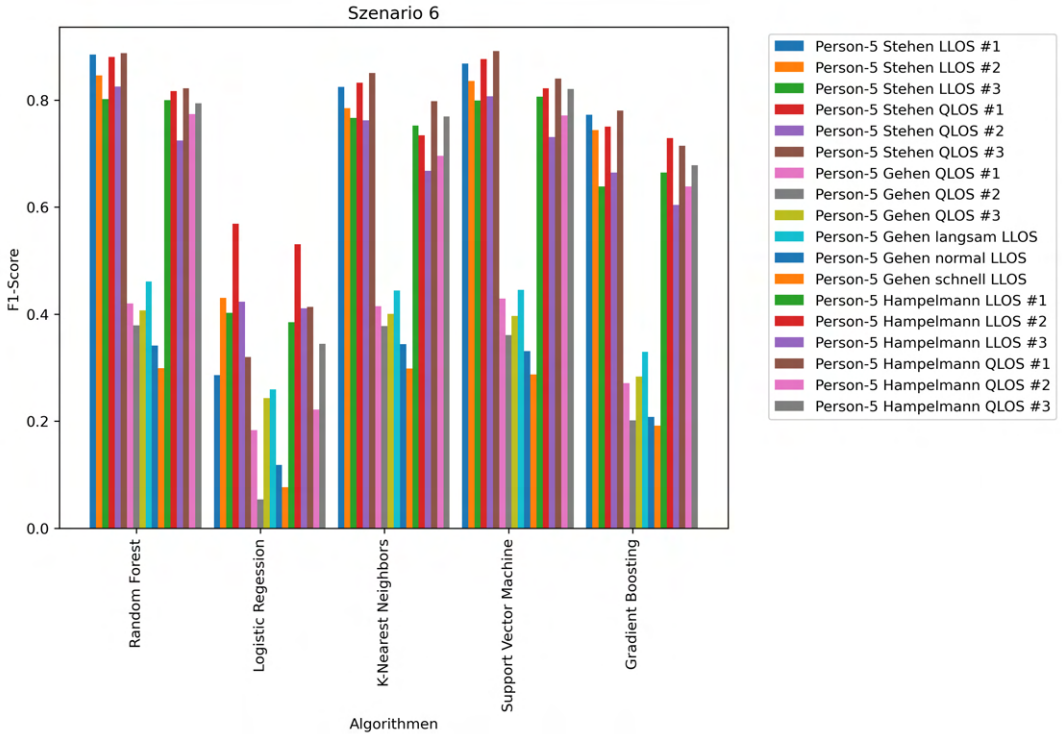


Abbildung 7.6: Ergebnis der Szenario 6 Analyse zur Unterscheidung von allen Bewegungen der Person 5.

Analyse 6:

Das letzte Szenario dieser Kategorie umfasst alle 18 Bewegungen der Person 5. Es lassen sich signifikante Unterschiede in den Ergebnissen der Algorithmen beobachten. Die Bewegung „Gehen“ unabhängig von der Position, Richtung oder Geschwindigkeit zeigt die schlechtesten Resultate. Dieser Trend konnte bereits beim Szenario 5 ebenfalls bei dieser Bewegung beobachtet werden. Der minimale F1-Score der Bewegung „Gehen QLOS #2“ von 0,05 beim Logistic Regression Algorithmus ist der niedrigste Wert aller Szenarien. Der Support Vektor Algorithmus weist die größte Spannweite von 0,6 bei einem maximalen F1-Score von 0,89 auf. Dieser Höchstwert wird ebenfalls vom Algorithmus Random Forest erreicht.

7.1.4 Unterscheidung von Bewegungen in Abhängigkeit von der Position

Im Rahmen dieser Untersuchung erfolgt eine Analyse der Differenzierbarkeit der Bewegungen an den verschiedenen Positionen. Die Betrachtung erfolgt an den Positionen #1, #2 und #3.

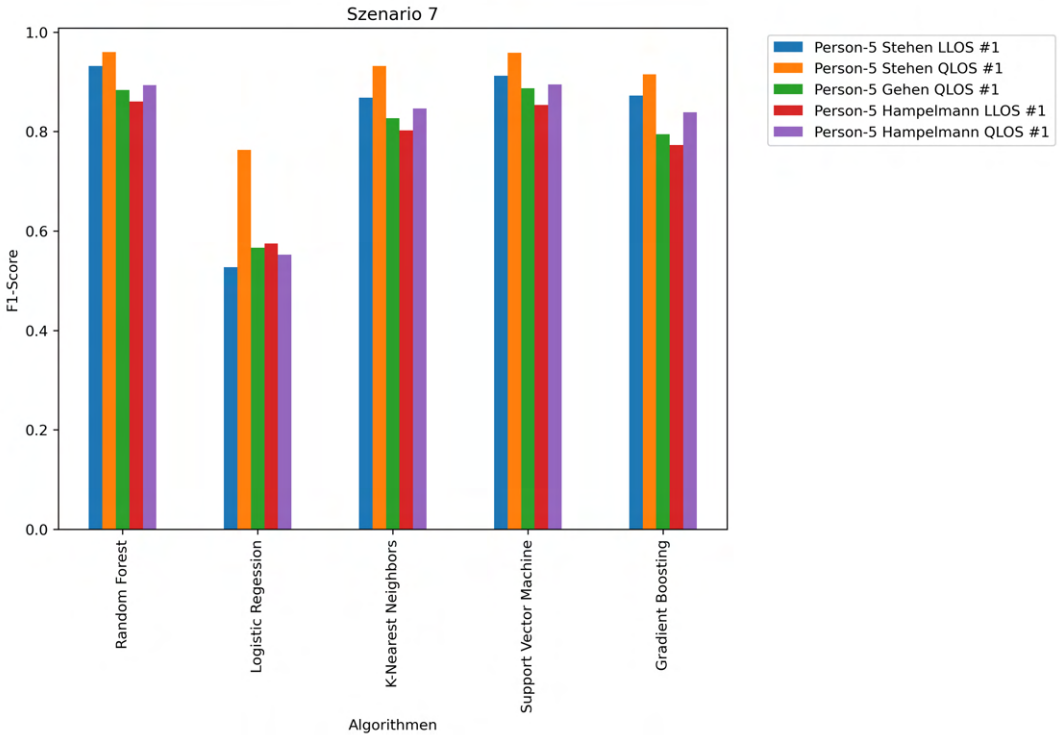


Abbildung 7.7: Ergebnis der Szenario 7 Analyse zur Unterscheidung von Bewegungen an der Position #1, die in Richtung LLOS und QLOS ausgeführt wurden.

Analyse 7:

In diesem Szenario wurden fünf verschiedene Bewegungen an der Position #1 durchgeführt. Der Algorithmus Random Forest erzielt mit einem durchschnittlichen F1-Score von 0,91 den höchsten Wert, gefolgt von der Support Vector Machine mit 0,9. Die Spannweite des F1-Scores dieser beiden Algorithmen ist nahezu identisch und liegt zwischen 0,85 und 0,96. Der Algorithmus Logistic Regression weist mit einem F1-Score von 0,6 wiederum den geringsten Wert auf. Allerdings weist er für die Bewegung „Stehen QLOS #1“ einen hohen F1-Wert von 0,76 auf. Dieser Höchstwert wird lediglich bei diesem Szenario in Verbindung mit dieser Bewegung erzielt. Eine entsprechende Tendenz ist ebenfalls bei den übrigen Algorithmen zu beobachten, wenngleich in einem geringeren Ausmaß.

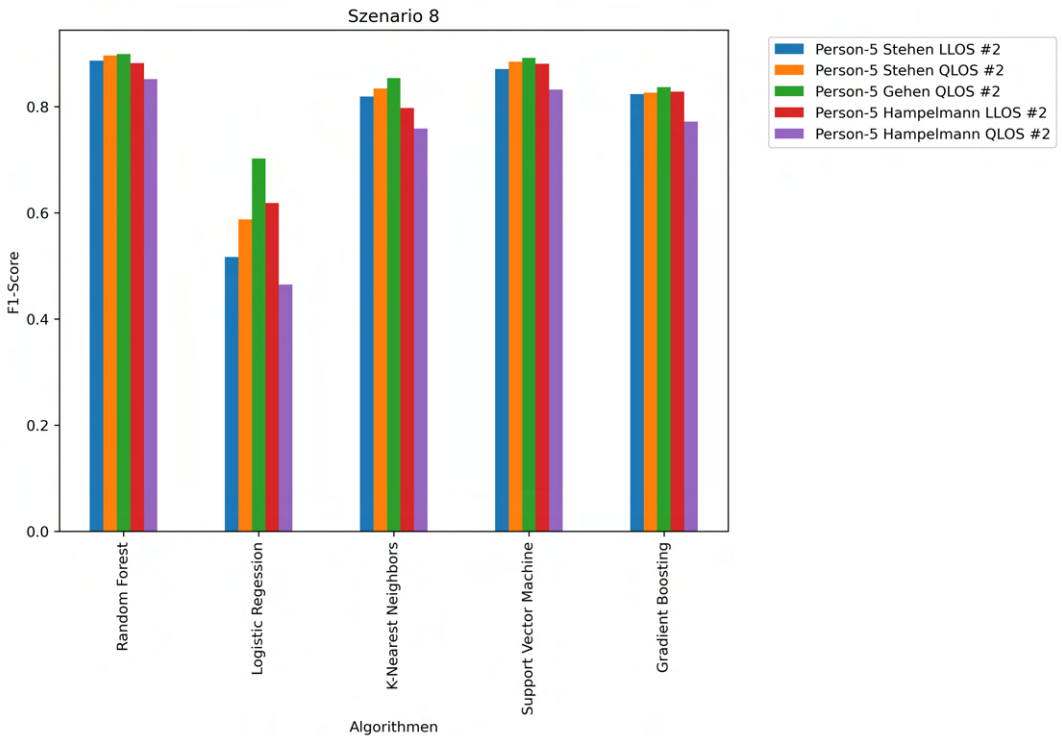


Abbildung 7.8: Ergebnis der Szenario 8 Analyse zur Unterscheidung von Bewegungen an der Position #2, die in Richtung LLOS und QLOS ausgeführt wurden.

Analyse 8:

In diesem Szenario werden die Bewegungen an der Position # 2 durchgeführt. Es handelt sich um die gleichen Bewegungen und ebenfalls um die Person 5 wie im Szenario 7. Tendenziell zeigt sich im Vergleich zum vorhergehenden Szenario 7, dass die Spannweite und die Standardabweichung bei allen Algorithmen, bis auf den Logistic Regression und den K-Nearest Neighbors, geringer sind. Ebenfalls ist im Vergleich ersichtlich, dass der hohe F1-Score der Bewegung „Stehen QLOS #1“ des Szenarios 7, beim Szenario 8 nicht mehr vorhanden ist. Anstatt dessen zeigt die Bewegung „Gehen QLOS #2“ den höchsten F1-Score von 0,7. Den höchsten F1-Score erreichen wiederum der Random Forest mit 0,88 und die Support Vector Machine mit 0,87. Die Minimal- und Maximalwerte dieses Scores unterscheiden voneinander nur gering.

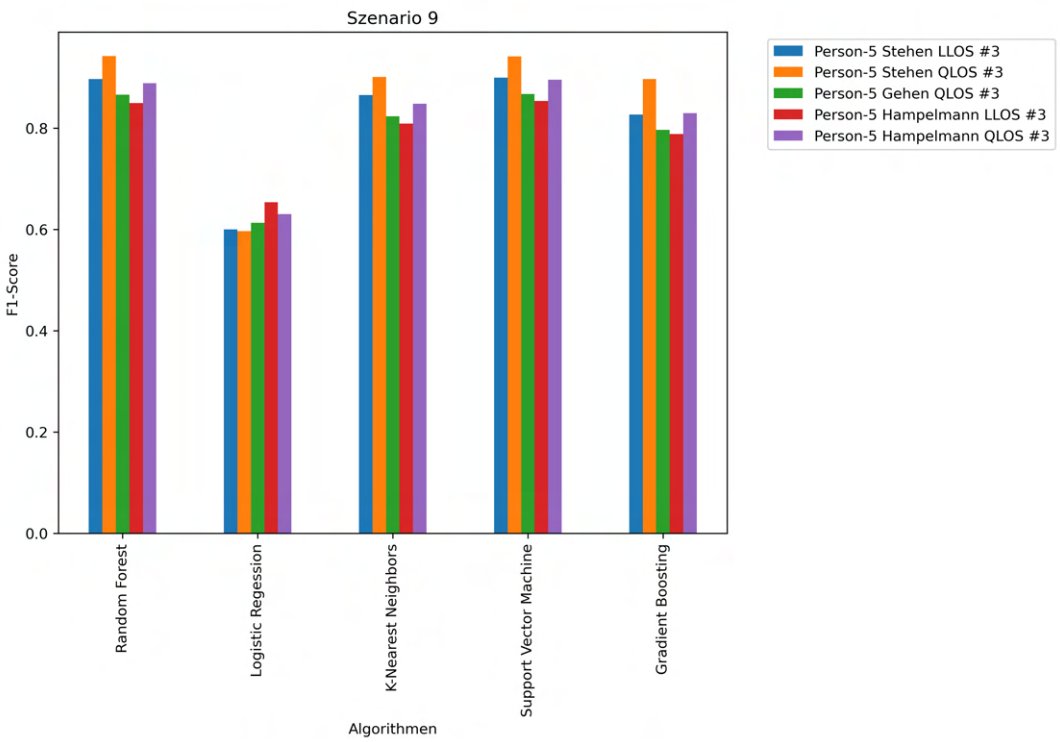


Abbildung 7.9: Ergebnis der Szenario 9 Analyse zur Unterscheidung von Bewegungen an der Position #3, die in Richtung LLOS und QLOS ausgeführt wurden.

Analyse 9:

In Szenario 9 erfolgt die Ausführung der Bewegung an Position #3. Die Analyse zeigt ein nahezu identisches Ergebnis wie bei in Szenario 7, mit Ausnahme der Logistic Regression. Der höchste F1-Score wurde wiederum mit 0,94 von den Algorithmen Random Forest und Support Vector Machine erzielt. Der K-Nearest Neighbor und die Gradient Boosting Machine liegen wieder im Mittelfeld der höchsten F1-Score Ergebnisse. Wie bereits in Szenario 7 festgestellt, weist die Bewegung „Stehen QLOS“ einen höheren F1-Score auf als die übrigen Algorithmen. Demgegenüber zeigt die Bewegung „Hamplemann LLOS“ den geringsten F1-Score. Der Algorithmus Logistic Regression hingegen liefert bei 4 von 5 gegenteilige Ergebnisse. Nur der „Hamplmann QLOS #3“ zeigt ähnliche Ergebnisse.

7.1.5 Positionsdetektion

In der letzten Kategorie erfolgte die Betrachtung der gleichen Person und Bewegung an unterschiedlichen Positionen. Dazu wurden alle Bewegungen pro Position zu einem neuen Datenset zusammengefasst, sodass drei Datensets mit jeweils den gleichen Bewegungen an unterschiedlichen Positionen resultierten.

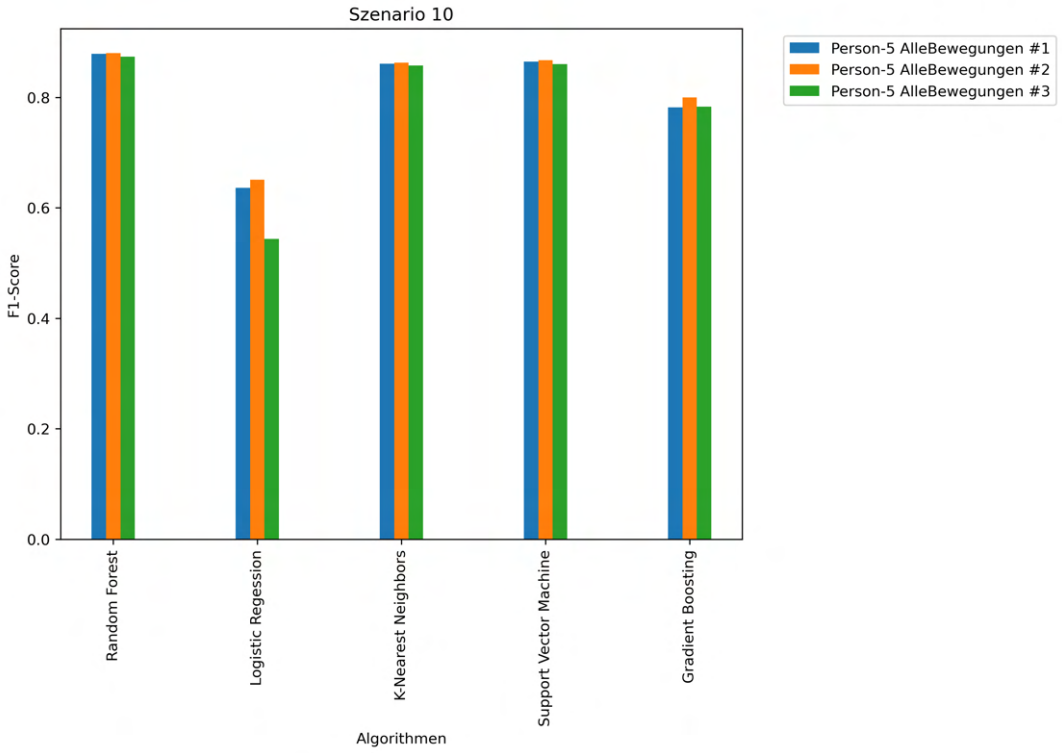


Abbildung 7.10: Ergebnis der Szenario 10 Analyse zur Unterscheidung von alle Bewegungen einer Person an verschiedenen Position.

Analyse 10:

Die Abbildung 7.10 zeigt die Resultate dieser Analyse. Der Algorithmus Random Forest weist mit einem F1-Score von 0,88 den höchsten Wert auf. Die beiden Algorithmen K-Nearest Neighbor und Support Vector Machine erzielen mit einem F1-Score von 0,86 nahezu identische Ergebnisse. Bei einem Vergleich des Szenarios 4 mit 10, bei dem ebenfalls drei Klassen unterschieden werden, zeigen sich beim maximalen F1-Score und der Spannweite Unterschiede. So liegt die Spannweite beim Szenario 10 zwischen 0,01 und 0,11 wohingegen dieser beim Szenario 4 0,16 bis 0,25 beträgt. Der maximale F1-Score des Szenario 10 ist im Durchschnitt um 0,2 höher als beim Szenario 4.

Die Auswertung der Machine-Learning-Ergebnisse der Szenarien zeigt, dass die Logistic Regression sehr schnell Ergebnisse liefert, der F1-Score jedoch gering ist. Im Gegensatz dazu liefern die Algorithmen Support Vector Machine und Gradient Boosting bessere Ergebnisse, die Laufzeit ist jedoch teilweise um den Faktor 100 höher als bei den anderen Algorithmen. Die Algorithmen Random Forest und K-Nearest Neighbor weisen eine ähnliche Genauigkeit wie die Support Vector Machine und der Gradient Boosting Algorithmus auf und dies bei einer deutlich kürzeren Laufzeit. Die Machine-Learning-Algorithmen wurden zum einen mit und zum anderen ohne die Dimensionsreduktion Principal Component Analysis durchgeführt. Dabei wurde der Wert *n_components* auf 30 gesetzt, was im Durchschnitt eine 95% erklärbare Spannweite bedeutet ([97]). Die Unterschiede zwischen den Ergebnissen der beiden Analysen betrugen $< 1\%$. Zusammenfassend kann festgestellt werden, dass die Unterscheidung von Personen, bei denen jede Person die gleiche Bewegung ausführt, für maschinelle Lernalgorithmen schwierig ist (F1-Score $< 0{,}52$). Dies ist bei den Szenarien 1 bis 3 der Fall. Dagegen können verschiedene Bewegungen, die von einer Person ausgeführt werden, besser unterschieden werden (Szenarien 4 bis 10). Es ist auffällig, dass die Bewegung „Gehen“ grundsätzlich die schlechteste Detektion aufweist. Vergleicht man hierzu die Anzahl der Datensätze der Szenarien, so ist kein Zusammenhang erkennbar, dass mehr Datensätze einen höheren Score ermöglichen. Es ist jedoch ersichtlich, dass die Laufzeit von der Anzahl der Datensamples abhängt. Die Tabelle zeigt, dass der F1-Score, von oben beginnend, mit fast jedem Szenario ansteigt. Dies legte die Vermutung eines „Data Leakage“ nahe. Aus diesem Grund wurden die Machine-Learning-Algorithmen zum Test in umgekehrter Reihenfolge durchlaufen, so dass ein Fehler im Skript ausgeschlossen werden konnte.

7.1.6 Use-Cases

Die durchgeführten Untersuchungen belegen, dass der Random Forest Algorithmus in sämtlichen Szenarien die besten Ergebnisse erzielt. Als zweitbeste Algorithmen können der K-Nearest Neighbor und der Gradient Boosting genannt werden, die bei den Szenarien 5, 7 bis 10 bzw. 7 bis 10 noch akzeptable Ergebnisse liefern. Die übrigen Algorithmen können nicht empfohlen werden, da entweder der F1-Score zu gering ist, wie dies bei der logistischen Regression der Fall ist, oder die Laufzeit zur Ermittlung des F1-Scores wesentlich höher liegt, als bei den anderen Algorithmen.

7.2 Erklärbarkeit

Im Rahmen der nachfolgenden Untersuchung zur Erklärbarkeit der Ergebnisse erfolgt zunächst eine visuelle Analyse, bevor im Anschluss Erklärbarkeitsmethoden zum Einsatz kommen. Die visuelle Darstellung zielt darauf ab, die Struktur der Kanalzustandsinformationen zu verdeutlichen und potenzielle Zusammenhänge zwischen den Datensätzen aufzuzeigen. Des Weiteren dient diese Darstellung der Ergebnisse der Erklärbarkeit. Im zweiten Schritt wird das Framework InterpretML verwendet, um

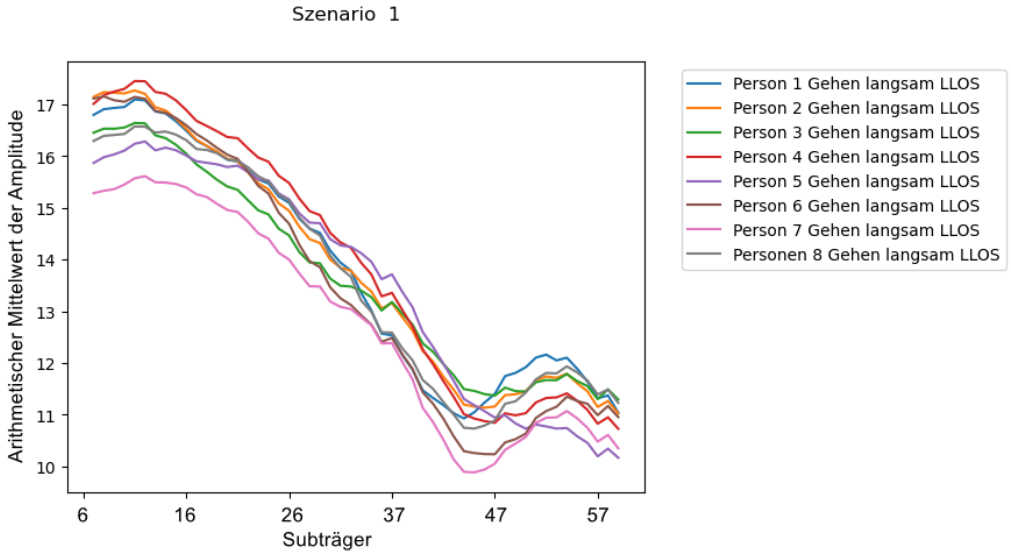


Abbildung 7.11: Darstellung des arithmetischen Mittelwerts der Amplitude pro Subträger für 8 Klassen.

die Explainable Boosting Machine anzuwenden. Anschließend wird in einem weiteren Schritt die Erklärbarkeitsmethode SHAP angewendet. Aufgrund der Tatsache, dass sowohl die Explainable Boosting Machine als auch SHAP in Verbindung mit Extreme Gradient Boosting eine Klassifizierung ermöglichen, wurde für diese ebenfalls eine Klassifikationsanalyse durchgeführt und in die Bewertung miteinbezogen.

7.2.1 Visuelle Erklärbarkeit

Nachfolgend wird untersucht, inwiefern sich Subträger identifizieren lassen, die einen besonderen Einfluss auf die Bewegungen haben, oder ob es andere Einflussfaktoren gibt, die dafür verantwortlich sind, dass Bewegungen unterschiedlich gut oder schlecht erkannt werden. Die Abbildung 6.2 zeigt die 3D-Ansicht der Bewegung „Person-1 Gehen langsam LLOS“ in der die Komplexität der Datenstruktur zu erkennen ist. Um eine einfachere visuelle Darstellung zu ermöglichen wurde in einer ersten Analyse der arithmetische Mittelwert jedes Subträgers des Szenario 1 gebildet und in einem Liniendiagramm dargestellt (Abbildung 7.11). Dieser Anwendungsfall beinhaltet die Bewegung „Gehen langsam LLOS“ von sieben verschiedenen Personen und einer Personengruppe mit zwei Personen. Es ist zu erkennen, dass die berechneten Werte pro Subträger einen ähnlichen Verlauf aufweisen. Darüber hinaus lässt sich feststellen, dass die einzelnen Klassen bei den Subträgern 6 bis 15 besser differenziert werden können als bei den übrigen Subträgern.

Im Rahmen eines Vergleichs wurde das Szenario 4 untersucht, um festzustellen, wie sich die Amplitudenwerte von unterschiedlichen Bewegungen der gleichen Person verhalten. Es zeigt sich ein ähnliches Bild wie beim Szenario 1 (Abbildung 7.12).

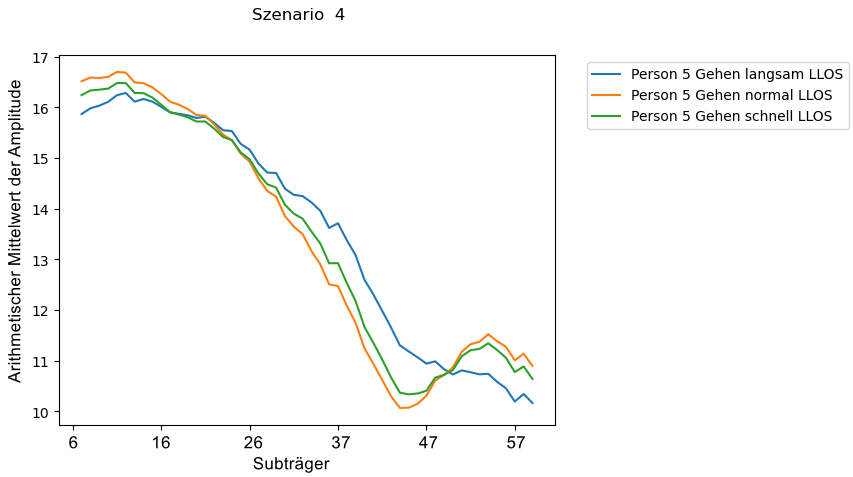


Abbildung 7.12: Darstellung des arithmetischen Mittelwerts der Amplitude pro Subträger (Feature) für drei Klassen.

Um die Unterschiede zwischen den Bewegungen in diesem Anwendungsfall deutlicher darzustellen, wurde eine Differenzbildung der errechneten Werte vorgenommen. Das Ergebnis der Analyse ist in der Abbildung 7.13 dargestellt. Es zeigt die Gemeinsamkeiten, sowie die Unterschiede bei den einzelnen Features (Subträger). Die Auswertung der Szenarien 2 und 3 sowie 5 bis 10 zeigt ebenfalls sehr ähnliche Ergebnisse.

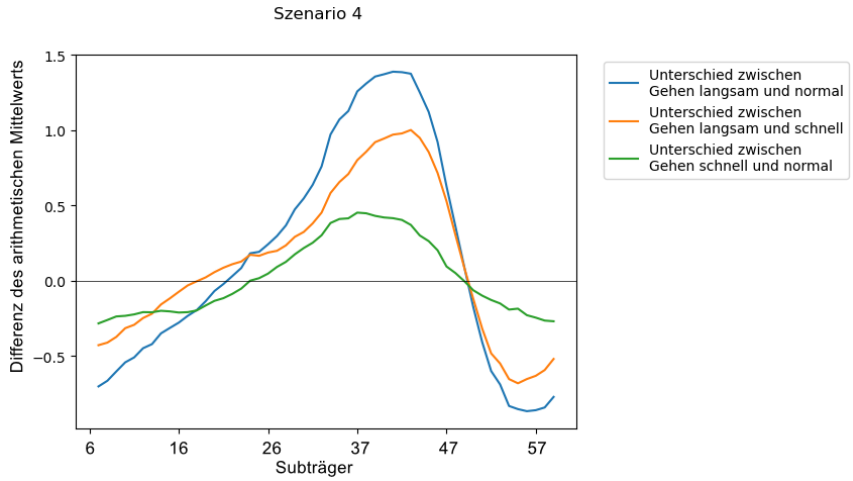


Abbildung 7.13: Die Abbildung veranschaulicht die Unterschiede der arithmetischen Mittelwerte des Szenarios 4. Die Differenzen sind insbesondere bei den Subträgern 35 bis 47 sowie 45 bis 57 deutlich erkennbar. Zudem sind der Schnittpunkt und die Nulldurchgänge beim Subträger 49 ersichtlich.

7.2.2 Explainable Boosting Machine

Die Explainable Boosting Machine (Abschnitt 4.1) kann sowohl für die Erklärbarkeit der Ergebnisse als auch für die Klassifikation verwendet werden. Um die Leistungsfähigkeit dieses Algorithmus mit der bereits durchgeführten Bewertung der Szenarien vergleichen zu können, wurde dieser im ersten Schritt ohne Optimierung der Hyperparameter auf die Datensätze der Szenarien 1 - 10 angewendet. Anschließend erfolgte die Bestimmung der Hyperparameter mittels HalvingGridSearch. Die Ermittlung der Parameter pro Szenario nahm einen Zeitraum von bis zu 18,7 Stunden in Anspruch. Zur Evaluierung wurde, wie bei den vorangehenden Analysen, der F1-Score herangezogen. Die Ergebnisse des F1-Scores mit und ohne Hyperparameteroptimierung sind in der Tabelle 8.16 dargestellt. Die Auswertung der Daten zeigt, dass der Unterschied des F1-Scores mit und ohne Hyperparameteroptimierung zwischen 0,01 und 10,09% liegt. Die Resultate der Explainable Boosting Machine mit Hyperparameteroptimierung sind in der Tabelle 7.2 dargestellt. Der F1-Score des Explainable Boosting Machine ist dem des Gradient Boosting Algorithmus vergleichbar. Allerdings zeigen sich signifikante Unterschiede in der Geschwindigkeit der beiden Algorithmen. Hier zeigt sich, dass die Explainable Boosting Machine deutlich schneller ist. Die Werte variieren zwischen 33-mal schneller für Szenario 6 bis zu 196-mal schneller für Szenario 4. In Bezug auf die Geschwindigkeit lässt sich die EBM grundsätzlich mit dem K-Nearest Neighbor Algorithmus vergleichen, wobei die EBM teilweise nochmal bis zu 13-mal schneller ist.

Tabelle 7.2: Ergebnisse des Explainable Boosting Machine Algorithmus mit optimierten Hyperparametern. Die benötigte Zeit für die Ermittlung des F1-Scores ist in der Spalte „Laufzeit“ angegeben. Die Angabe der Samples bezeichnet die Datensätze des jeweiligen Szenario.

Szenario	**Laufzeit (min)**	**Samples**	**Anzahl der Klassen**	**F1-Score**
1	20,65	371.150	8	0,37
2	19,92	374.083	8	0,36
3	17,77	370.736	8	0,34
4	0,78	144.197	3	0,51
5	112,13	704.932	15	0,68
6	168,12	849.128	18	0,57
7	6,37	236.284	5	0,83
8	6,9	238.245	5	0,80
9	6,04	230.405	5	0,82
10	113,62	704.922	3	0.77

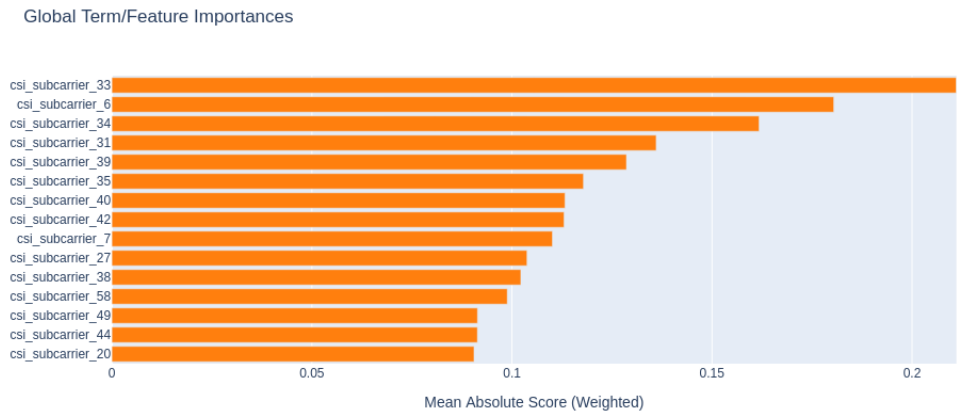


Abbildung 7.14: Globale Erklärbarkeit des Szenario 1 mit Explainable Boosting Machine. Auf der Y-Achse werden die Subträger dargestellt. Die X-Achse gibt den gewichteten mittleren absoluten Score der jeweiligen Merkmale an und gibt Auskunft über deren Wichtigkeit. Dies erfolgt durch das Messen der durchschnittlichen absoluten Auswirkung eines Merkmals auf die Modellvorhersagen.

Zur Ermittlung der globalen Erklärbarkeit wurde in einem ersten Schritt mit den Framework InterpretML der Explainable Boosting Machine-Algorithmus auf das Szenario 1 angewendet. Die wichtigsten Features (Feature Importances) zeigt die Abbildung 7.14.

Im weiteren Verlauf erfolgt die Betrachtung der Szenarien 1 bis 3 für die Bewegung „Gehen langsam, normal und schnell“.

Von den 15 in Ergebnissen dargestellten Subträgern, werden acht mit dem höchst gewichteten mittleren absoluten Score (Weighted Mean Absolute Score - WMAS) betrachtet, der das relevanteste Unterscheidungsmerkmal darstellt. EBM verwendet diesen Score zur Bewertung der Wichtigkeit von Merkmalen. Es zeigt sich, dass die Merkmale zur Unterscheidung der Bewegungen zwischen den Szenarien differieren. Die Tabelle 7.3 präsentiert die Ergebnisse dieser Auswertung. Zur besseren Kennzeichnung wurden die Felder farblich markiert. Die in der Tabelle grün markierten Felder geben an, dass diese Subträger in allen Klassen unter den acht besten Werten vorhanden sind. Wird ein Subträger nur zweimal aufgeführt, so erfolgt die Kennzeichnung durch eine gelbe Markierung. Bei einem einmaligen Vorkommen wird die Kennzeichnung durch eine rote Markierung vorgenommen.

Zur Ermittlung der Subträger mit den höchsten gewichteten mittleren absoluten Score erfolgte für die Szenarien 4 bis 6 zur Unterscheidung von Bewegungen und 7 bis 9 (Unterscheidung von Personen in Abhängigkeit der Position) ebenfalls eine Analyse. Für das Szenario 10 kann aufgrund der Datenlage eine Analyse nicht erfolgen, sodass in der Bewertungstabelle (Tabelle 7.6) nur der ermittelte Score angegeben wird.

Tabelle 7.3: Die Unterscheidung der Personen der Szenarien 1 bis 3 erfolgt anhand der gewichteten mittleren absoluten Scores (WMAS). Dabei zeigt sich, dass die Subträger 6, 31, 33 und 34 unter den vier Subträgern den höchsten Score aufweisen. Ein zweimaliges Vorkommen ist hingegen bei zwei Subträgern gegeben. Ein einmaliges Vorkommen zeigt sich bei vier Subträgern.

Szenario	**Laufzeit (Min.)**								
1	40,01	Subträger	33	6	34	31	39	35	40	42
		WMAS	0,21	0.18	0,16	0,13	0,12	0,12	0,11	0,11
2	38,86	Subträger	6	58	33	31	49	34	40	57
		WMAS	0,2	0.19	0,15	0,12	0,11	0,11	0,10	0,10
3	38,50	Subträger	6	33	34	31	39	58	38	40
		WMAS	0,21	0,21	0,13	0,13	0,12	0,11	0,10	0,10

Tabelle 7.4: Die folgende Tabelle präsentiert die wichtigsten Merkmale zur Unterscheidung von Bewegungen der Szenarien 4 bis 6. Es zeigt sich, dass lediglich zwei Merkmale (6 und 46) in allen drei Bewegungen vorkommen. Die Anzahl der Subträger, die zweimal vorkommen, hat sich hingegen auf sechs erhöht. Das Vorhandensein von nur einem Merkmal kommt sechsmal vor.

Szenario	**Laufzeit (Min.)**								
4	1,18	Subträger	44	45	51	43	46	6	50	52
		WMAS	0,08	0,07	0,07	0,07	0,06	0,06	0,06	0,06
5	229	Subträger	58	57	47	46	42	43	6	48
		WMAS	0,97	0,49	0,46	0,46	0,45	0,45	0,44	0,42
6	356	Subträger	58	57	6	46	47	48	39	42
		WMAS	0,84	0,42	0,42	0,39	0,39	0,36	0,35	0,35

Im Folgenden wird eine Auswertung präsentiert, welche die Szenarien 7 bis 9 zur Unterscheidung von Personen in Abhängigkeit von deren Position betrachtet.

Tabelle 7.5: Die dargestellte Tabelle zeigt die wichtigsten Merkmale zur Unterscheidungen von Bewegungen der Szenarien 7 bis 9. Das Merkmal 58 repräsentiert bei den jeweiligen Szenarien jeweils den höchsten gewichteten mittleren absoluten Score. Demgegenüber kommen fünf Subträger zweimal und elf Subträger einmal vor.

Szenario	**Laufzeit (Min.)**									
7	1,18	Subträger	58	57	56	55	33	30	34	31
		WMAS	1,18	0,63	0,50	0,37	0,30	0,29	0,29	0,28
8	229	Subträger	58	6	57	47	56	7	48	24
		WMAS	0,54	0,51	0,40	0,28	0,26	0,25	0,25	0,24
9	356	Subträger	58	28	29	30	40	47	41	31
		WMAS	0,43	0,32	0,31	0,30	0,30	0,30	0,30	0,29

Tabelle 7.6: Die folgende Tabelle zeigt die wichtigsten Merkmale zur Unterscheidungen von Bewegungen in Abhänigkeit von der Position des Szenario 10. Im Gegensatz zur vorherigen Auswertung sind in diesem Szenario die Subträger 6 und 58 , die den höchsten Score darstellen, vertauscht.

Szenario	**Laufzeit (Min.)**									
10	27	Subträger	6	58	42	20	19	43	13	33
		WMAS	0,27	0,20	0,18	0,16	0,16	0,16	0,15	0,14

Eine Gesamtbetrachtung der Ergebnisse über alle zehn Szenarien zeigt die Abbildung 7.15. Hierbei wurde die Häufigkeit der Subträger aller Szenarien zusammengefasst und als Balkendiagramm dargestellt. Detaillierte Informationen der einzelner Subträger können mit Hilfe des interaktiven Visualisierungsdashboard von InterpretML dargestellt werden. Hierzu wurde beispielhaft der Subträger 33 ausgewählt, der im Szenario 1 das wichtigste Merkmal darstellt (Abbildung 7.16).

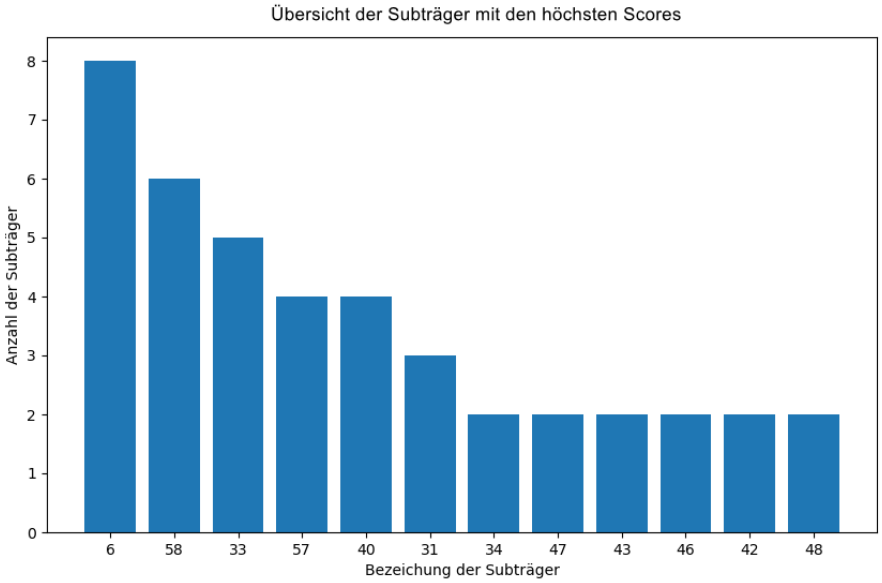


Abbildung 7.15: Übersicht der Subträger mit dem höchsten Score nach deren Anzahl. Hierbei ist ersichtlich, dass der Subträger 6 am häufigsten vorkommt, gefolgt vom Subträger 58, 33, 57 usw. Aus Gründen der Übersichtlichkeit erfolgt nur ein Darstellung von 12 Subträgern.

Die lokale Erklärbarkeit mittels des Explainable Boosting Machine-Algorithmus erlaubt die Betrachtung der Auswirkungen eines einzelnen Datensamples auf die Vorhersagen des Modells. Als Beispiel wurde ein Datensamaple des Szenarios 1 ausgewählt, der die Klasse 64 vorhersagt. Diese entspricht der Bewegung der „Person 7 Gehen langsam“. Die Abbildung 7.17 zeigt die interaktive Ausgabe von InterpretML. Betrachtet man den Subträger 34, so leistet dieser beispielsweise zur Klasse 1 einen Beitrag zur Vorhersage von 0,032, zur Klasse 19 einen Beitrag von 0,5, zur Klasse 25 einen Beitrag von 0,63 und zur Klasse 67 einen Betrag von 0,45. Der Intercept-Wert stellt einen konstanten Wert dar, der als Basis für die Berechnung der Vorhersage dient. Der lokale Vorhersagewert wird durch Hinzufügen des Gewichts berechnet. Das Szenario 1 beinhaltet 371.150 Datensamples und 52 Merkmalen, weshalb nur eine Klasse exemplarisch zur Darstellung der lokalen Erklärbarkeit aufgeführt wurde. Wie in Abbildung 7.17 dargestellt, resultiert die Multiklassifikation in Verbindung mit den 52 Merkmalen in einer Vielzahl von Abhängigkeiten. Die Verwendung von 15 beziehungsweise 18 Klassen bei den Szenarien 5 und 6 verdeutlicht die starke Abhängigkeit der Subträger untereinander.

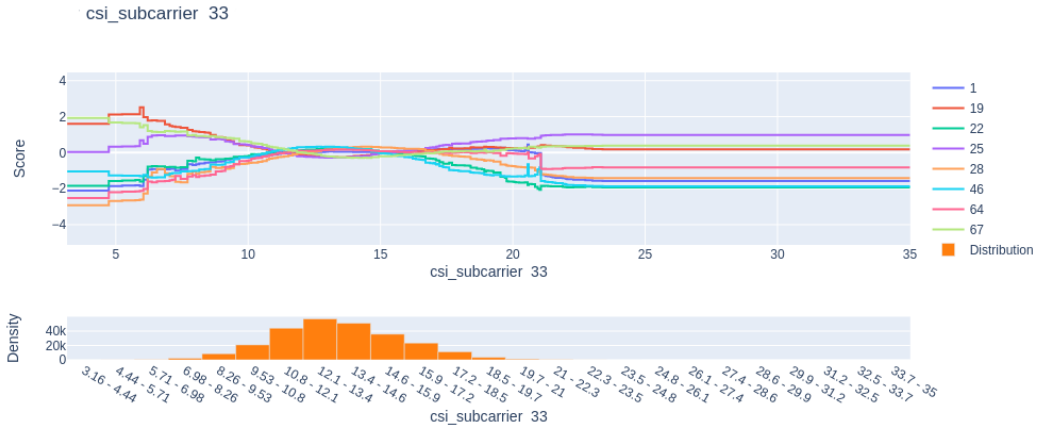


Abbildung 7.16: Die Darstellung der lokalen Erklärbkarkeit des Subträger 33 mit EBM des Szenario 1. Der untere Bereich zeigt die Dichte der Datenverteilung des Subträgers. Im oberen Bereich der Abbildung werden auf der rechten Seite die Klassen des Szenarios 1 präsentiert. Auf der X-Achse ist der Bereich aufgeführt, in dem sich die Amplitudenwerte der Subträger bewegen. Die Y-Achse gibt den Wert der Einflussnahme auf die Vorhersage an.

7.2.3 eXtreme Gradient Boosting

Im Folgenden wird die Erklärbarkeit von Szenarien mit SHAP untersucht. Die Post-hoc-Methode berechnet SHAP-Werte, um die Erklärbarkeit von Modellen zu ermöglichen. Zur Ermittlung der SHAP-Werte wurde hierzu in einem ersten Schritt der Machine-Learning-Algortihmus Random Forest angewendet. Der initiale Test zur Ermittlung der SHAP-Werte hat gezeigt, dass der Algorithmus innerhalb eines Zeitraums von 24 Stunden beim kleinsten Datenset kein Resultat lieferte. Dieses Szenario 4 umfasst 144.199 Datensamples und 52 Merkmale. Aus diesem Grund wurde der Extreme Gradient Boosting Algorithmus eingesetzt, der für die Berechnung der Ergebnisse unter anderem die Grafikprozessoreinheit (GPU) verwenden kann. Der Extreme Gradient Boosting Algorithmus, mit aktivierter GPU-Unterstützung, lieferte für das Szenario 1 mit 371.257 Datensamples innerhalb von 7 Sekunden ein Ergebnis. Um einen Vergleich des Explainable Boosting Machine und Extreme Gradient Boosting Algorithmus zu ermöglichen, erfolgt ebenfalls die Ermittlung der Hyperparameter mittels HalvingGridSearch. Die Ergebnisse mit und ohne Hyperparameter sind in der Tabelle 8.18 dargestellt. Durch die Hyperparameteroptimierung verbessert sich der F1-Score um bis zu 6%. Es sei darauf hingewiesen, dass der Extreme Gradient Boosting eine fortlaufende Zahl als Label benötigt, die bei 0 beginnt. Dies hatte zur Konsequenz, dass die Klassen vor der Verwendung entsprechend angepasst werden mussten.

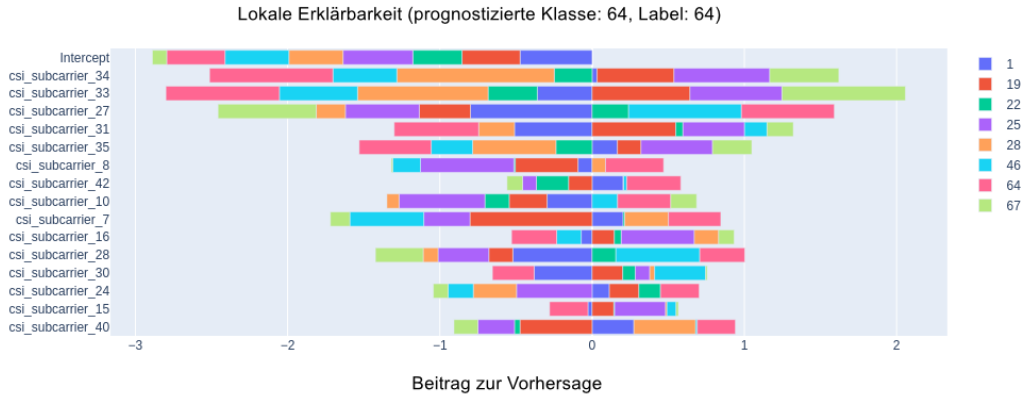


Abbildung 7.17: Darstellung der lokalen Erklärbkarkeit des Subträger 42 des Szenarios 1. Auf der Y-Achse ist der Intercept-Wert dargestellt, sowie die Subträger dieses Szenarios. Die Breite des jeweiligen Plots zeigt den Beitrag zur Vorhersage der prognostizierten Klasse. Die positiven und negativen Werte des Beitrags stellen die unterschiedliche Einflussnahme der Merkmale auf die Zielvariable dar. Der Intercept-Wert veranschaulicht den Basiswert der Vorhersage der für jede Klasse unterschiedlich ist.

Die ermittelten Hyperparameter sind in der Tabelle 8.19 aufgeführt, die zur Ermittlung des F1-Scores Verwendung fanden. Der Extreme Gradient Boosting Algorithmus weist ähnliche Ergebnisse wie der Random Forest Algorithmus auf, wobei die Differenz zwischen den F1-Scores zwischen 0,9 und 3,6 % liegt. Des Weiteren ist die Geschwindigkeit des Extreme Gradient Boosting Algorithmus mit der des Logistic Regression Algorithmus vergleichbar.

Tabelle 7.7: F1-Score des XGBoost Algorithmus mit optimierten Hyperparametern. Die sehr kurzen Laufzeiten sind aufgrund der Verwendung der GPU-Unterstützung möglich.

Szenario	**Laufzeit (Sek.)**	**Samples**	**Anzahl der Klassen**	**F1-Score**
1	7	371.150	8	0,50
2	6	374.083	8	0,47
3	6	370.736	8	0,44
4	2	144.197	3	0,57
5	18	704.932	15	0,80
6	27	849.128	18	0,69
7	3	236.284	5	0,92
8	3	238.245	5	0,90
9	3	230.405	5	0,91
10	7	704.922	3	0.87

7.2.4 SHAP

Die lokale Erklärbarkeit wird mit SHAP allgemein durch den Wasserfall-Plot dargestellt und erklärt. Dieser beschreibt, um welchen Wert das jeweilige Merkmal zur Abweichung der Vorhersage vom Basiswert beigetragen hat. Aufgrund der hohen Anzahl an Informationen erwies sich diese Art der Darstellung jedoch als nicht empfehlenswert. Aus diesem Grund erfolgte die Darstellung der Ergebnisse mittels Heatmaps, die nachfolgend dargestellt werden. Die Y-Achse der Headmap gibt hierbei die Bezeichnung der Subträger an. Im mittleren Bereich der Y-Achse wurde bewusst der Subträger 32 entfernt, da es sich hierbei um einen Nullträger handelt und für die Ermittlung der Ergebnisse keine Relevanz besitzt. Auf der X-Achse sind die Klassen aufgeführt, die den Bewegungen zugeordnet sind.

Die Gesamtbetrachtung der Ergebnisse über alle zehn Szenarien wird in der Abbildung 7.18 dargestellt. Es lässt sich feststellen, dass die Subträger 6 und 58 am häufigsten auftreten, gefolgt vom Subträger 7, 43, 44 und 45. Ab dem Subträger 34 finden sich diese maximal dreimal in den Gesamtergebnissen, weshalb eine weitere Darstellung an dieser Stelle unterbleibt.

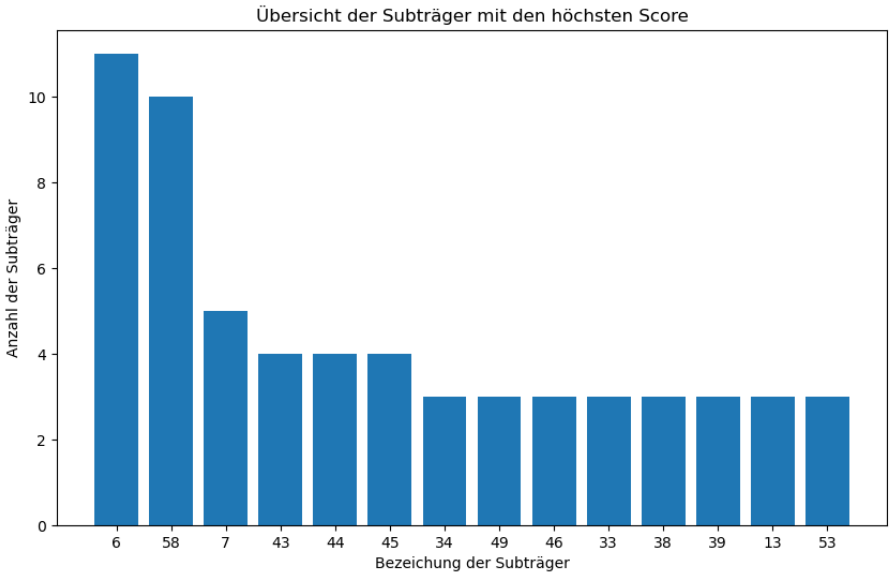
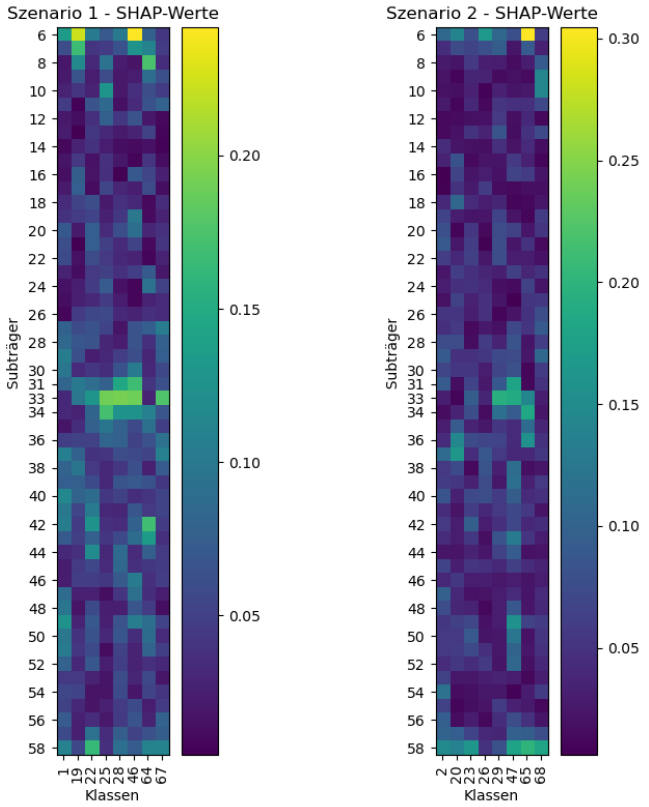


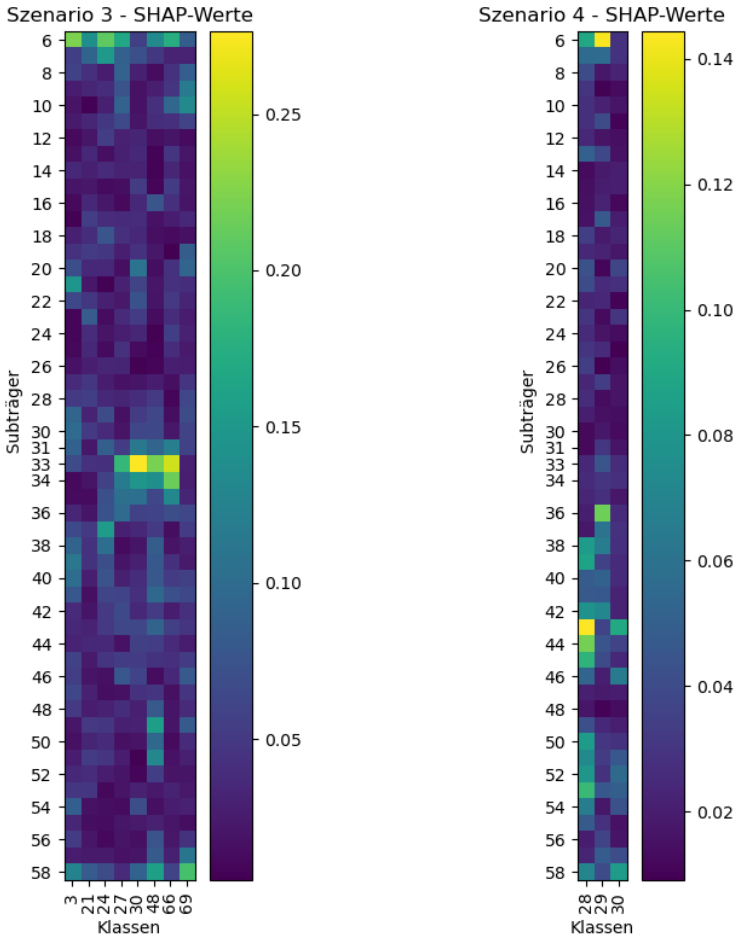
Abbildung 7.18: Übersicht der Subträger mit dem höchsten Score (Quelle: eigene Darstellung)



(a) Gehen langsam (b) Gehen normal

Abbildung 7.19: Darstellung der SHAP-Werte für der Bewegungen „Gehen langsam und normal“ von allen Personen und der Personengruppe. Links dargestellt die Ergebnisse des Szenario 1 und rechts die des Szenarios 2.

Analyse 1 und 2: Die Abbildung 7.19 präsentiert die ermittelten SHAP-Werte für die Bewegung „Gehen von allen Personen in den Geschwindigkeiten langsam und normal“. Das Szenario 1 „Gehen langsam“ (links) zeigt eine Clusterbildung der Subträger 30 bis 34 für alle Klassen mit Ausnahme von „Person-1 Gehen langsam LLOS“ (Klasse 1) und „Person-7 Gehen langsam LLOS“ (Klasse 64). In diesem Bereich bewegen sich die SHAP-Werte zwischen 0,15 und 0,2. Der Subträger 6 zeigt bei fast allen Klassen, mit Ausnahme der Bewegung „Personen-8 Gehen langsam LLOS“ (Klasse 67), einen SHAP-Wert von über 0,10. Bei der Bewegung „Person-6 Gehen LLOS langsam“ (Klasse 46) liegt dieser bei 0,23. Eine ähnliche Darstellung zeigt die rechte Abbildung der Bewegung „Gehen normal“. Auch hier ist eine Clusterbildung bei den Subträgern 31 bis 34 zu erkennen. Ebenso ist ein erhöhter SHAP-Wert von 0.3 beim Subträger 6 der Bewegung „Person-7 Gehen normal LLOS“ (Klasse 65) festzustellen. Zudem zeigt der Subträger 58 höhere SHAP-Werte im Bereich von 0,15 bis 0,20 an.



(a) Gehen schnell

(b) Gehen langsam, normal und schnell

Abbildung 7.20: Links dargestellt die SHAP-Werte der Bewegungen „Gehen schnell“ von allen Personen und der Personengruppe. In der rechten Abbildung sind die SHAP-Wert der Bewegung „Gehen langsam, normal und schnell“ der Person 5 zu sehen.

Analyse 3 und 4: Die Abbildung 7.20 zeigt das Szenario 3 und 4. Die Skalierung dieses Szenarios unterscheidet sich geringfügig um 0,04 von den Szenarien 1 und 2. Auch hier stellen die bereits genannten Subträger die höchsten SHAP-Werte von 0,15 bis 0,3 dar. Die rechte Abbildung stellt das Szenario 4 mit den Bewegungen „Gehen langsam, normal und schnell“ von einer Person mit einer geringeren Skalierung bis 0,14 dar. Die Subträger 31, 33 und 34 sind nicht mehr, wie bei den Szenarien 1 bis 3, so dominant. Im Gegensatz dazu weisen die Subträger 36 bis 58 SHAP-Werte zwischen 0,08 und 0,14 auf.

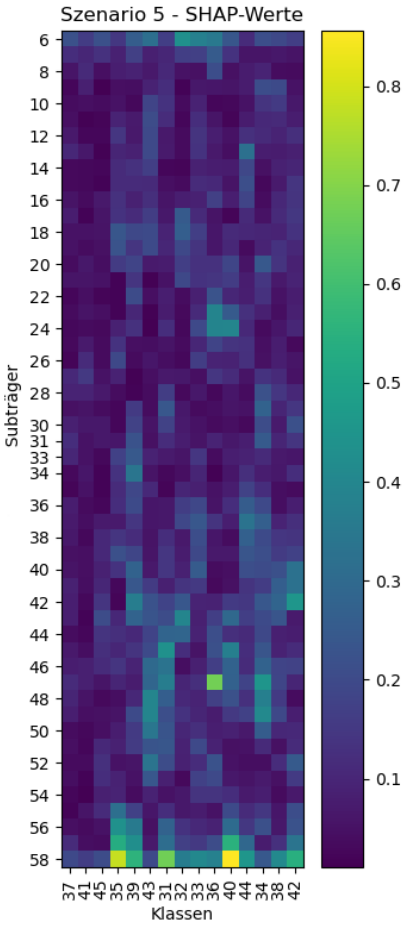


Abbildung 7.21:
Szenario 5 mit allen Bewegungen außer „Gehen langsam, normal und schnell“ der Person 5 mit 15 Klassen.

Analyse 5: Im Rahmen des Szenario 5, welches in der Abbildung 7.21 dargestellt ist, erfolgte die Analyse von 15 Klassen. Diese beinhalten alle Bewegungen außer „Gehen langsam, normal und schnell“ einer Person. Die Skalierung der SHAP-Werte von 0,1 bis 0,8 weist gegenüber Szenarien 1 bis 3 einen um den Faktor 2,6 höheren Wertebereich auf. Die Subträgernummern 57 und 58 weisen die höchsten SHAP-Werte auf, die zwischen 0,4 und 0,8 liegen.

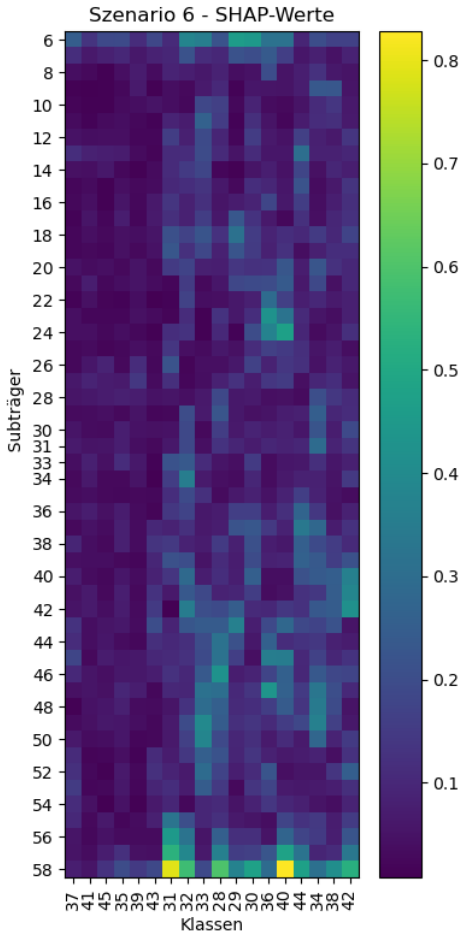
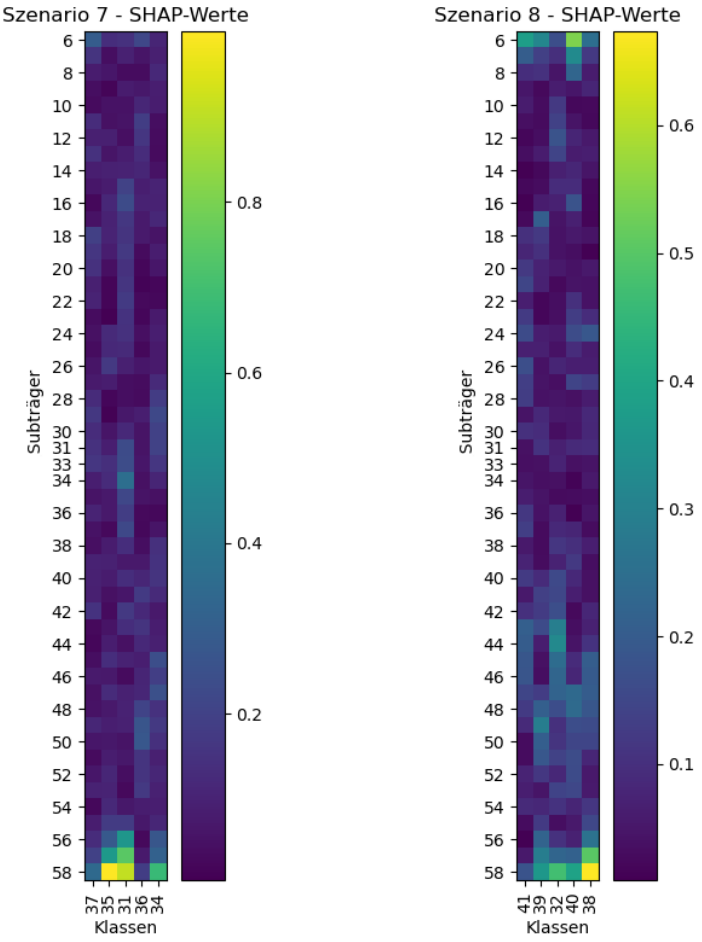


Abbildung 7.22:
SHAP-Werte des Szenario 6 mit allen Bewegungen der Person 5 mit 18 Klassen.

Analyse 6: Die Abbildung 7.22 stellt das Szenario 6 mit der gleichen Skalierung wie das Szenario 5 dar und beinhaltet alle Bewegungen der Person 5. Es zeigt sich ein horizontales Clustering, das beim Subträger 43 beginnt. Eine derartige Ausprägung ist bei den andern Szenarien (1 - 5) nicht ersichtlich. Auch hier manifestiert sich ein teilweise höherer SHAP-Wert von 0,4 bis 0,8 ab den Subträgern 55.

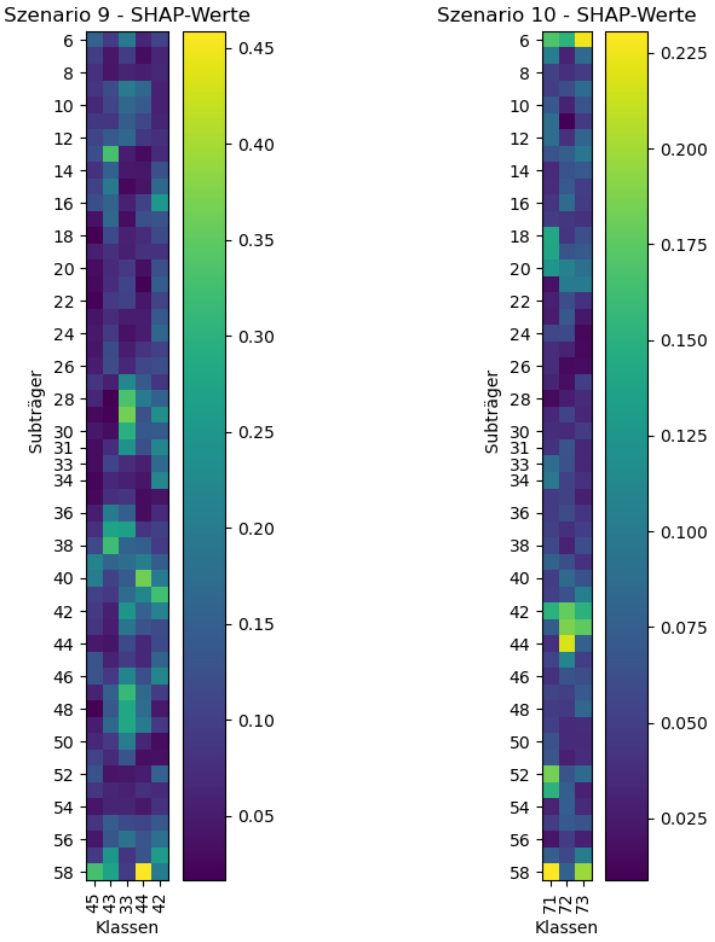


(a) Position #1

(b) Position #2

Abbildung 7.23:
Szenario 7 und 8 mit allen Bewegungen der Person 5 an Position #1 und #2.

Analyse 7 und 8: Das Szenario 7, dargestellt in der linken Abbildung, umfasst alle Bewegungen der Person 5 an der Position #1. Die größten SHAP-Werte reichen von 0,5 bis 0,88 und sind bei den Subträgern 56 bis 58 zu finden. Die rechte Seite dieser Abbildung präsentiert das Resultat des Szenarios 8, das sich vom Szenario 7 in der Position (#2) und der Skalierung unterscheidet. Die Subträger zeigen, wie beim vorhergehenden Szenario, bei den Subträgern 56 bis 58 SHAP-Werte von 0,4 bis 0,6.



(a) Position #3

(b) kombinierte Bewegungen

Abbildung 7.24: SHAP-Werte für der Szenarien 9 und 10

Analyse 9 und 10: Die letzte Abbildung 7.24 zeigt die Szenarien 9 und 10. Die linke Abbildung zeigt das Szenario 9 mit allen Bewegungen an der Position #3 einer Person. Im Gegensatz zu den Szenarien 7 und 8, bei denen die gleichen Bewegungen lediglich an anderen Positionen durchführt werden, zeigt diese Abbildung geringere SHAP-Werte. Ab dem Subträger 27 lässt sich eine vertikale Clusterbildung erkennen. Die rechte Abbildung zeigt die Zusammenfassung der bereits beschriebenen Bewegungen. Bei den Subträgern 6, 42 bis 44 und 58 zeigen sich höhere SHAP-Werte.

7.3 Bewertung

Die visuelle Bewertung zeigt, dass die Detektion, welche Subträger eine Unterscheidung von Bewegungen ermöglicht, auf diese Weise nicht direkt ermittelt werden kann. Des Weiteren ist nicht ersichtlich, wie sich der Kurvenverlauf (Abbildung 7.11 und 7.12), der durch die Amplitudenwerte bestimmt wird, ergibt. Ein anderes Bild stellt sich durch die Verwendung von Explainable Boosting Machine und SHAP dar. Das Visualisierungsdashboard, welches durch InterpretML bereitgestellt wird, ermöglicht eine interaktive Darstellung von lokalen und globalen Vorhersagen. Auf diese Weise kann der Betrag für jedes Merkmal (Subträger) zur Vorhersage dargestellt werden. Die globale Erklärbarkeit hingegen ermöglicht die Darstellung der Merkmale, die für die Ergebnisse ausschlaggebend sind. Diese „Feature Importances“ werden in einem Boxplot entsprechend der Wichtigkeit dargestellt. Verwendet wurde hierzu der Machine-Learing-Algorithmus Explainable Boosting Machine. Zur Erklärbarkeit eines Blackbox-Modells wurde SHAP in Verbindung mit dem Machine-Learning-Algorithmus Extreme Gradient Boosting verwendet. Letzterer ermöglicht die Berechnung der SHAP-Werte durch die Verwendung des Grafikprozessors (GPU) mit hoher Geschwindigkeit. Bei der Analyse der SHAP-Werte lassen sich einige Subträger identifizieren, die einen Einfluss auf die Detektion der Bewegungen haben. Zu den identifizierten Subträgern zählen die Subträger 6, 7, 29 bis 34 und 57 bis 58. Darüber hinaus lassen sich einzelne Clusterbildungen von Subträgern erkennen, die jedoch nicht eindeutig einer bestimmten Bewegung zugeordnet werden können. Eine übergeordnete Betrachtung zeigt zudem, dass ab dem Subträger 27 bis 58 höhere und divergentere SHAP-Werte zu verzeichnen sind. Ein direkter Zusammenhang von Subträger zu bestimmten Bewegungen ist jedoch nicht erkennbar. Bei einem Vergleich der Ergebnisse von SHAP und Explanable Boosting Maschine wird ersichtlich, dass die Subträger 6, 58, 33, 34, 43 und 46 bei beiden Algorithmen eine hohe Relevanz für die Erklärbarkeit aufweisen. Durch die Zusammenführung der wichtigsten Merkmale von EBM und SHAP stellen sich folgende Subträger als die Wichtigsten heraus:

Tabelle 7.8: Subträger mit der größten Wichtigkeit für die Vorhersage eines Modells, absteigend sortiert nach der Anzahl des Auftretens.

Subträger																		
6	58	33	34	43	46	57	7	40	44	31	44	47	49	42	39	48	13	53

8 Fazit und Ausblick

In diesem Kapitel erfolgt die Zusammenfassung der wichtigsten Ergebnisse dieser Arbeit. Im Anschluss wird dargestellt, welche weiteren Forschungsarbeiten im Bereich des Machine-Learning und der Erklärbarkeit der Ergebnisse durchgeführt werden könnten.

8.1 Fazit

Die Analyse von Kanalzustandsinformationen ermöglicht die Erkennung von Bewegungen, die vielseitige Möglichkeiten eröffnet. Gezeigt wurde in dieser Arbeit, dass mit Hilfe von technischen Hilfsmitteln Bewegungsinformationen erfasst und aufgezeichnet werden können. Für die Weiterverarbeitung der gespeicherten Daten stehen verschiedene Hilfsmittel wie beispielsweise die verwendete Software-Bibliothek scikit-learn zur Verfügung, die für die Verwendung von Machine-Learing-Algorithmen konzipiert wurden. Die Verarbeitung und Auswertung der aufgezeichneten Daten erfolgte mit Python und den Machine-Learning-Algorithmen Random Forest, Logistic Regression, K-Nearest-Neighbor, Support Vector Machine, Gradient Boosting Machine, XGBoost und Explainable Boosting Machine.

Die Ergebnisse der Klassifikation zeigen, dass die Unterschiede zwischen den verwendeten Algorithmen meist nur gering sind. Eine Ausnahme stellt der Logistic Regression Algorithmus dar, der bei allen Analysen den schlechtesten F1-Score lieferte. Die Ursache hierfür ist vermutlich in der nicht-linearen Beziehung zwischen den unabhängigen Merkmalen zu finden. Die besseren Ergebnisse des Random Forest könnten ein Hinweis auf diese These sein. Des Weiteren zeigt sich bei allen Machine-Learning-Algorithmen, dass die Bewegung „Gehen“ grundsätzlich schlecht klassifiziert werden kann. Dies könnte auf die Aufbauhöhe des Senders und Empfängers zurückzuführen sein, sodass die Bewegungen unterhalb dieser Höhe nur geringe Änderungen der Kanalzustandsinformationen hervorrufen.

Bei der Betrachtung der Laufzeit, die für die Ermittlung der Klassifikationsergebnisse benötigt wurde, zeigt der Algorithmus Logistic Regression die kürzeste Laufzeit. Im Gegensatz dazu benötigt die Support Vektor Machine am längsten zur Ermittlung der Ergebnisse. Der Random Forest und K-Nearest-Neighbor zeigen bei der Gesamtbetrachtung die beste Performance in Bezug auf Zeit und Genauigkeit, gefolgt von der Gradient Boosting Machine. Dabei ist zu bedenken, dass der K-Nearest-Neighbor als Lazy Learner immer bessere Werte in Bezug auf die Zeit erzielt, da er in der Trainingsphase kein Modell lernt.

Bei der Anwendung der Erklärbarkeitsmethoden erfolgt ebenfalls eine Klassifizierung mit den Explainable Boosting und dem XGBoost Algorithmus. Im Rahmen eines Vergleichs dieser beiden Machine-Learning-Algorithmen zeigt sich, dass der XGBoot-Algorithmus mit kürzester Laufzeit die Ergebnisse ausgibt. Bei einem Vergleich zu den Algorithmen Random Forest, Logistic Regression, K-Nearest-Neighbor, Support Vector Machine und Gradient Boosting Machine kann festgestellt werden, dass auch in diesem Fall der XGBoost Algorithmus innerhalb der kürzesten Zeit die Ergebnisse liefert. Zurückzuführen ist dies auf die verwendete Grafikprozesseinheit der NVIDIA Grafikkarte mit einer hohen Anzahl an CUDA-Kernen. Dies erlaubt die Ausführung einer Vielzahl von Rechenoperationen gleichzeitig, was zu einer Beschleunigung der Ermittlung der Ergebnisse führt.

Die Relevanz eines Merkmals für die Vorhersagewerte eines Modells wird mit InterpretML und SHAP ermittelt. Die Darstellung der Ergebnisse mit InterpretML erfolgt mit einem interaktiven Dashboard. Sowohl die lokale als auch die globale Erklärbarkeit kann übersichtlich dargestellt werden. Die ermittelten Ergebnisse sind nicht persistent. Es ist jedoch möglich diese als JSON-File zu speichern und für spätere Analysen zu verwenden. Für die Darstellung der Ergebnisse mit SHAP wurde pro Szenario eine Headmap erstellt. Diese zeigt prinzipiell die gleichen Informationen wie InterpretML. Die lokale Erklärbarkeit, die auf Instanzebene (Datenpunkt) die Wichtigkeit für die Vorhersage bestimmt, beinhaltet sehr viele Informationen, die für diesen Anwendungsfall als ungeeignet erscheinen. Eine globale Erklärbarkeit, die hingegen auf Features (Merkmalen) basiert, ermöglicht die Darstellung der Wichtigkeit der Merkmale. Die mit EBM und SHAP ermittelten wichtigsten Subträger sind in der Tabelle 7.8 dargestellt. Bei den unterschiedlichen Szenarien zeigt sich jedoch, dass nicht immer der gleiche Subträger den entsprechenden Beitrag zur Ermittlung des Ergebnisses liefert. Es wurde kein Subträger identifiziert, der für jedes Szenario den größten Beitrag zur Ermittlung des Ergebnisses liefert.

8.2 Ausblick

Die nachfolgenden Themenpunkte sind als Zusammenfassung zu sehen, die während der Anfertigung der Arbeit entstanden sind. Zu einigen Punkten gibt es bereits wissenschaftliche Arbeiten, die sich teilweise mit dem Thema beschäftigt haben.

Themenpunkt 1:
In dieser Arbeit erfolgte nur die Berücksichtigung der Amplitudenwerte. Die Kanalzustandsinformationen stellen weitere Daten zur Verfügung die für Unterscheidung von Bewegungen verwendet werden können. Aus den Real- und Imaginärdaten kann zusätzlich die Phase des Antennensignals berechnet werden. Sodass die Möglichkeit besteht, entweder nur die Phase oder in Kombination mit der Amplitude diese für die Unterscheidung von Bewegungen zu verwenden ([98]).

Themenpunkt 2:
Zusätzlich zu den Real- und Imaginärwerten erfolgt die Übertragung von RSSI-Daten (Received Signal Strength Indicator). Diese Daten stellt die Empfangsfeldstärke einer Funkverbindung dar. Dadurch, dass diese Daten ebenfalls von Umwelteinflüssen beeinflusst werden, können Positionsortungen erfolgen [99]. In Kombination mit den Kanalzustandsinformationen bietet dies eventuell die Möglichkeit bessere Unterscheidungen von Bewegungen zu ermöglichen.

Themenpunkt 3:
In der Vorverarbeitungsphase dieser Arbeit erfolgte die Anwendung des Hampelfilters. Dieser wurde verwendet, um Ausreißer in den Amplitudenwerten zu entfernen. Nicht betrachtet wurde jedoch die Tatsache, dass Rauschen in den Signalamplitude die Ergebnisse beeinflussen können. Die Verwendung der diskrete Wavelet-Transformation (Discrete Wavelet Transform - DWT) ermöglicht beispielsweise das Entfernen von hochfrequenten Rauschen in den Datensets ([100]). Es gibt noch weitere Filter die Verwendung finden können.

Themenpunkt 4:
Die Verwendung von Neuronalen Netzen könnte zu besseren Ergebnissen führen. Im Vergleich zu den hier verwendeten Machine-Learning-Algorithmen haben Neuronale Netze den Vorteil, Muster in den Daten besser erkennen zu können. Als Beispiel für ein Neuronales Netz, das für die Zeitreihenanalyse geeignet ist, kann das „Convolutional Neural Network (CNN)“ in Verbindung mit „Data Augmentation (DA)“ genannt werden ([101]).

Themenpunkt 5:
Die in den Szenarien verwendeten Bewegungen unterscheiden sich von jenen, die bei der Eingabe von Benutzernamen und Passwörtern ausgeführt werden. Die Bewegungen der Finger, wie beispielsweise bei der Verwendung mit dem Smartphone komplexer. Um diese feingranularen Bewegungen präzise erfassen zu können, ist es eventuell notwendig die Positionen der Sender- und Empfangsgeräte zu überdenken.

Themenpunkt 6:
Mit Hilfe der Erklärbarkeitsalgorithmen wurden CSI-Subträger ermittelt, die für Wichtigkeit der Vorhersagen der Machine-Lerning-Modelle eine hohe Relevanz darstellen. Es sollte verifiziert werden, in welchen Ausmaß sich dies auf die unterschiedlichen Algorithmen auswirkt. Zudem zeigt die Bewegung „Gehen“ bei allen verwendeten Algorithmen ein schlechteres Ergebnis, als die übrigen Bewegungen. Eventuell kann mit den Ergebnissen der Erklärbarkeitsmethoden eine Hinweis auf die Ursache gefunden werden.

Literatur

[1] Carsten Roppel. *Grundlagen der Nachrichtentechnik*. 2023. URL: https://www.hanser-elibrary.com/doi/book/10.3139/9783446478831 (besucht am 01. 06. 2024).

[2] Mo Li. *Atheros CSI tool*. 2022. URL: https://wands.sg/research/wifi/AtherosCSI/ (besucht am 01. 06. 2024).

[3] Mohammed Al-qaness u. a. „Channel State Information from Pure Communication to Sense and Track Human Motion: A Survey“. In: *Sensors* 19.15 (2019), S. 3329. ISSN: 1424-8220. URL: https://www.mdpi.com/1424-8220/19/15/3329.

[4] Andrii Zhuravchak. „Human Activity Recognition based on Wi-Fi CSI Data -A Deep Neural Network Approach“. In: *Procedia Computer Science* 198 (2022), S. 59–66. ISSN: 1877-0509. URL: https://www.sciencedirect.com/science/article/pii/S1877050921024509.

[5] Shen Xingfa u. a. *WiPass: 1D-CNN-based smartphone keystroke recognition Using WiFi signals*. 2021. URL: https://www.sciencedirect.com/science/article/pii/S1574119221000523 (besucht am 01. 06. 2024).

[6] Yuxi Wang u. a. „WiFall Device-Free Fall Detection by Wireless Networks“. In: *IEEE Transactions on Mobile Computing* 16.2 (2017), S. 581–594. DOI: 10.1109/TMC.2016.2557792.

[7] SpringerLink. *Grundkurs Mobile Kommunikationssysteme*. 2023. URL: https://link.springer.com/book/10.1007/978-3-658-36963-7 (besucht am 01. 06. 2024).

[8] Martin Bossert. *Einführung in die Nachrichtentechnik*. München: Oldenbourg Verlag, 2012. ISBN: 978-3-486-71744-0. DOI: 10.1524/9783486717440. URL: https://www.degruyter.com/document/doi/10.1524/9783486717440/html.

[9] G Laurent. *Radio modulation*. 2023. URL: https://www.sqimway.com/modulation.php (besucht am 01. 06. 2024).

[10] Carsten Rpooel. *Grundlagen der Nachrichtentechnik*. 2023. URL: https://www.hanser-elibrary.com/doi/book/10.3139/9783446478831 (besucht am 01. 06. 2024).

[11] Thilo Manske. *3.3 Modulation*. 2012. URL: http://einstein.informatik.uni-oldenburg.de/lehre/semester/seminar/02ss/wlan/WLANnode9.html (besucht am 01. 06. 2024).

[12] SpringerLink. *Nachrichtentechnik*. 2023. URL: https://link.springer.com/book/10.1007/978-3-8348-9742-8 (besucht am 01. 06. 2024).

[13] Keysight Technologies. *Standard (802.11a/g/j/p OFDM)*. 2023. URL: https://rfmw.em.keysight.com/wireless/helpfiles/89600B/WebHelp/Subsystems/wlan-ofdm/content/dlg_ofdm_fmt_standard.htm (besucht am 01. 06. 2024).

[14] Mathworks. *OFDM*. 2023. URL: https://de.mathworks.com/discovery/ofdm.html (besucht am 01. 06. 2024).

[15] Karl-Dirk Kammeyer u. a. *Nachrichtenübertragung*. 6., erweiterte und aktualisierte Auflage. Lehrbuch. Wiesbaden und Heidelberg: Springer Vieweg, 2018. ISBN: 9783658170042.

[16] Joachim Meyer. *HW/SW Co-Design Framework für Hochgeschwindigkeits-OFDM Signalverarbeitung*. 2014. URL: https://publikationen.bibliothek.kit.edu/1000045040/3381995.

[17] Robert Klinski. *Referenzdatenfreie Kanalschätzung für Multiträgerübertragung*. Hrsg. von Technische Universität München. 2002. URL: https://mediatum.ub.tum.de/doc/601532/601532.pdf.

[18] Prof. Dr.-Ing. Dietmar Rudolph. *Inter-Symbol-Interferenz und Nequist-Bedingung*. Hrsg. von Dietmar Rudolph. 2006. URL: https://www.diru-beze.de/funksysteme/skripte/DiFuSy_S06/DiFuSy_ISI_SS06.pdf.

[19] Yidong Lang. *Das OFDM-Multitragerverfahren*. 2012. URL: https://www.ant.uni-bremen.de/sixcms/media.php/102/10753/top.pdf (besucht am 01. 06. 2024).

[20] E. Ghayoula u. a. „Capacity and Performance of MIMO systems for Wireless Communications“. In: *Journal of Engineering Science and Technology Review* 7.3 (2014), S. 108–111. ISSN: 17919320. DOI: 10.25103/jestr.073.17.

[21] Soukaena Hashem u. a. „Critical and Important Factors Related with Enhancing Wireless Communication Using MIMO Technology“. In: *Diyala Journal of Engineering Sciences* 08 (Apr. 2015), S. 42–63. DOI: 10.24237/djes.2015.08104.

[22] Karl-Dirk Kammeyer. *Nachrichtenübertragung*. 3., neubearbearbeite und ergänzte Auflage. Informationstechnik. Wiesbaden und s.l.: Vieweg+Teubner Verlag, 2004. ISBN: 978-3-519-26142-1. URL: https://link.springer.com/book/10.1007/978-3-322-94062-9.

[23] Biljana Baldic. „Space-Time Block Coding for Multiple Antenna Systems“. dissertation. Technischen Universität Wien, 2005. URL: https://publik.tuwien.ac.at/files/pub-et_10819.pdf.

[24] Keysight Technologies. *Concepts of Orthogonal Frequency Division Multiplexing (OFDM) and 802.11 WLAN)*. 2023. URL: https://rfmw.em.keysight.com/wireless/helpfiles/89600B/WebHelp/Subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm (besucht am 01. 06. 2024).

[25] Jialin Liu u. a. „Multi-Target Intense Human Motion Analysis and Detection Using Channel State Information“. In: *Sensors* 18.10 (2018). ISSN: 1424-8220. DOI: 10.3390/s18103379. URL: https://www.mdpi.com/1424-8220/18/10/3379.

[26] P. Tejera u. a. „Feedback of Channel State Information in Wireless Systems“. In: *IEEE International Conference on Communications, 2007*. Piscataway, NJ: IEEE Operations Center, 2007, S. 908–913. ISBN: 1-4244-0353-7. DOI: 10.1109/ICC.2007.154.

[27] Yalong Xiao und other. „Exploiting distribution of channel state information for accurate wireless indoor localization“. In: *Computer Communications* 114 (2017), S. 73–83. ISSN: 0140-3664. URL: https://www.sciencedirect.com/science/article/abs/pii/S0140366417304152.

[28] David Tse u. a. *Fundamentals of wireless communication*. Cambridge: Cambridge University Press, 2005. ISBN: 9780511807213. DOI: 10.1017/CBO9780511807213.

[29] Maksim Lopatin u. a. „Object Classification Based on Channel State Information Using Machine Learning“. In: Jan. 2021, S. 367–374. ISBN: 978-3-030-58867-0. DOI: 10.1007/978-3-030-58868-7_42.

[30] Yichuan Zhang u. a. „Indoor Seat Occupancy Classification with Wi-Fi Channel State Information and Machine Learning Methods“. In: *2021 4th International Conference on Sensors, Signal and Image Processing*. ACM Digital Library. New York,NY,United States: Association for Computing Machinery, 2021, S. 72–81. ISBN: 9781450385725. DOI: 10.1145/3502814.3502826.

[31] Juan Niu und orthers. *CSI-F: A Human Motion Recognition Method Based on Channel-State-Information Signal Feature Fusion*. 2024. DOI: 10.3390/s24030862. (Besucht am 01. 06. 2024).

[32] Andreas Holzinger u. a. „Explainable AI Methods - A Brief Overview“. In: Apr. 2022, S. 13–38. ISBN: 978-3-031-04082-5. DOI: 10.1007/978-3-031-04083-2_2.

[33] Ioannis Belle Vaishak und Papantonis1. *Principles and Practice of Explainable Machine Learning*. 2021. DOI: 10.3389/fdata.2021.688969. (Besucht am 01. 06. 2024).

[34] Europäisches Parlament. *Was ist künstliche Intelligenz und wie wird sie genutzt?* 2024. URL: https://www.europarl.europa.eu/topics/de/article/20200827STO85804/was-ist-kunstliche-intelligenz-und-wie-wird-sie-genutzt (besucht am 01. 06. 2024).

[35] kobold. *Machine Learning vs. AI*. 2024. URL: https://www.kobold.ai/ml-vs-ai/ (besucht am 01. 06. 2024).

[36] SpringerLink. *Knowledge Science – Grundlagen*. 2023. URL: https://link.springer.com/book/10.1007/978-3-658-41689-8 (besucht am 01. 06. 2024).

[37] Scikit-learn Developers. *Multiclass and multioutput algorithms*. 13. Apr. 2024. URL: https://scikit-learn.org/stable/modules/multiclass.html (besucht am 01. 06. 2024).

[38] Ramprabhu Sreekrishnan. *Demystifying K-fold Cross Validation*. 2023. URL: https://medium.com/@prabhucs01/demystifying-k-fold-cross-validation-3c9d410adddd (besucht am 01. 06. 2024).

[39] Matthias Feurer und Frank Hutter. „Hyperparameter Optimization“. In: *Automated Machine Learning: Methods, Systems, Challenges*. Hrsg. von Frank Hutter, Lars Kotthoff und Joaquin Vanschoren. Cham: Springer International Publishing, 2019, S. 3–33. ISBN: 978-3-030-05318-5. DOI: 10.1007/978-3-030-05318-5_1.

[40] Gianluca Malato. *Hyperparameter tuning. Grid search and random search*. 19. Mai 2021. URL: https://www.yourdatateacher.com/2021/05/19/hyperparameter-tuning-grid-search-and-random-search/ (besucht am 01. 06. 2024).

[41] scikit-learn. *Support Vector Machines*. 2023. URL: https://scikit-learn.org/stable/modules/svm.html (besucht am 01. 06. 2024).

[42] Stefan Papp u. a. *Handbuch Data Science und KI: Mit Machine Learning und Datenanalyse Wert aus Daten generieren*. 2. Auflage. Hanser eLibrary. München: Hanser, 2022. ISBN: 9783446472457. DOI: 10.3139/9783446472457.

[43] Ehsan Harirchian u. a. „A Machine Learning Framework for Assessing Seismic Hazard Safety of Reinforced Concrete Buildings“. In: *Applied Sciences* 10 (Okt. 2020). DOI: 10.3390/app10207153.

[44] Jamal Amani Rad. *Learning with fractional orthogonal kernel classifiers in support vector machines: Theory, algorithms and applications*. Hrsg. von Jamal Amani Rad, Kourosh Parand und Snehashish Chakraverty. Industrial and applied mathematics. Singapore: Springer, 2023. ISBN: 9789811965524. URL: https://link.springer.com/book/10.1007/978-981-19-6553-1.

[45] Jason Brownlee. *One-vs-Rest and One-vs-One for Multi-Class Classification*. 2021. URL: https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/ (besucht am 01. 06. 2024).

[46] Michael Jordan u. a. *The Kernel Trick*. 2004. URL: http://www.cs.berkeley.edu/~jordan/courses/281B-spring04/lectures/lec3.pdf (besucht am 01. 06. 2024).

[47] springerprofessional.de. *Hands-on Machine Learning with Python*. 2022. URL: https://www.springerprofessional.de/hands-on-machine-learning-with-python/20195344 (besucht am 01. 06. 2024).

[48] Namita Mutha. *Bernoulli Naive Bayes*. 2023. URL: https://iq.opengenus.org/bernoulli-naive-bayes/ (besucht am 01. 06. 2024).

[49] Max Kuhn und otherss. *Applied predictive modeling*. Corrected at 5th printing. New York: Springer, 2016. ISBN: 9781461468493. URL: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=6245972.

[50] De Gruyter. *Practical AI for Business Leaders, Product Managers, and Entrepreneurs*. 2023. URL: https://www.degruyter.com/document/doi/10.1515/9781501505737/html (besucht am 01. 06. 2024).

[51] Dr. Mathias Jesussek. *Logistische Regression*. 2023. URL: https://datatab.de/tutorial/logistische-regression (besucht am 01. 06. 2024).

[52] Arun Addagatla. *Maximum Likelihood Estimation in Logistic Regression*. 2023. URL: https://arunaddagatla.medium.com/maximum-likelihood-estimation-in-logistic-regression-f86ff1627b67 (besucht am 01. 06. 2024).

[53] Aurelien Geron. *Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow: Konzepte, Tools und Techniken für intelligente Systeme*. 2. Auflage. Heidelberg: O'Reilly, 2020. ISBN: 9783960103394. URL: https://www.oreilly.com/library/view/praxiseinstieg-machine-learning/9781492065838/.

[54] Jörg Frochte. *Maschinelles Lernen*. Carl Hanser Verlag GmbH, 2018. ISBN: 9783446452916. URL: https://www.hanser-elibrary.com/isbn/9783446452916.

[55] Baochang Zhang und other. *Machine Learning and Visual Perception*. 1. Auflage. Berlin/Bosten: De Gruyter, 2020. ISBN: 978-3-11-059556-7. DOI: 10.1515/9783110595567.

[56] SpringerLink. *An Introduction to Statistical Learning*. 2023. URL: https://link.springer.com/book/10.1007/978-1-4614-7138-7 (besucht am 01. 06. 2024).

[57] Davis David. *Random Forest Classifier Tutorial: How to Use Tree-Based Algorithms for Machine Learning*. 2020. URL: https://www.freecodecamp.org/news/how-to-use-the-tree-based-algorithm-for-machine-learning/ (besucht am 01. 06. 2024).

[58] Tianqi Chen und Carlos Guestrin. „XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Aug. 2016. DOI: 10.1145/2939672.2939785. (Besucht am 01. 06. 2024).

[59] Data Science. *Boosting-Algorithmen – AdaBoost, Gradient Boosting, XGBoost*. 2022. URL: https://datascientest.com/de/gradient-boosting-algorithmen (besucht am 01. 06. 2024).

[60] Sebastian Raschka u. a. *Machine Learning mit Python*. 2023. URL: https://www.mitp.de/IT-WEB/KI-Data-Science/Machine-Learning-mit-Python.html (besucht am 01. 06. 2024).

[61] Sebastian Raschka. *STAT 479: Machine LearningLecture Notes*. 2018. URL: https://sebastianraschka.com/pdf/lecture-notes/stat479fs18/02_knn_notes.pdf (besucht am 01. 06. 2024).

[62] Cord Technologies. *Confusion Matrix*. 2023. URL: https://encord.com/glossary/confusion-matrix/ (besucht am 01. 06. 2024).

[63] Khalid K. Al-jabery u. a. „Computational Learning Approaches to Data Analytics in Biomedical Applications". In: *Computational Learning Approaches to Data Analytics in Biomedical Applications* (2023). URL: https://www.sciencedirect.com/book/9780128144824/computational-learning-approaches-to-data-analytics-in-biomedical-applications (besucht am 01. 06. 2024).

[64] Tahmid Z. Chowdhury. „Using Wi-Fi Channel State Information (CSI) for HumanActivity Recognition and Fall Detection". dissertation. THE UNIVERSITY OF BRITISH COLUMBIA (Vancouver), 2018. URL: https://open.library.ubc.ca/media/stream/pdf/24/1.0365967/4.

[65] Miguel Otero Pedrido. *Hampel*. 2023. URL: https://github.com/MichaelisTrofficus/hampel_filter (besucht am 01. 06. 2024).

[66] Amanda Casari Zheng Alice Zheng und Casari. *Merkmalskonstruktion für Machine Learning: Prinzipien und Techniken der Datenaufbereitung*. 2024. URL: https://www.oreilly.com/library/view/merkmalskonstruktion-fur-machine/9781492072089/ (besucht am 01. 06. 2024).

[67] Christian Reinboth. *sklearn.preprocessing.RobustScaler*. 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html (besucht am 01. 06. 2024).

[68] Ashwin Sharma. *StandardScaler, MinMaxScaler and RobustScaler techniques – ML*. 2023. URL: https://www.geeksforgeeks.org/standardscaler-minmaxscaler-and-robustscaler-techniques-ml/ (besucht am 01. 06. 2024).

[69] PythonProg. *Scikit-Learn's preprocessing.normalize in Python*. 2023. URL: https://www.pythonprog.com/sklearn-preprocessing-normalize/ (besucht am 01. 06. 2024).

[70] Alexandre Gramfort u. a. *data.py*. 2024. URL: https://github.com/scikit-learn/scikit-learn/blob/51a765a/sklearn/preprocessing/data.py#L1295 (besucht am 01. 06. 2024).

[71] Joachim Steinwendner u. a. *Neuronale Netze programmieren mit Python*. 2020. URL: https://www.rheinwerk-verlag.de/neuronale-netze-programmieren-mit-python/ (besucht am 01. 06. 2024).

[72] Ian T. Jolliffe. *Principal component analysis*. 2. ed., [Nachdr.] Springer series in statistics. New York, Berlin und Heidelberg: Springer, 2004. ISBN: 0387954422. URL: http://cda.psych.uiuc.edu/statistical_learning_course/Jolliffe%20I.%20Principal%20Component%20Analysis%20(2ed.,%20Springer,%202002)(518s)_MVsa_.pdf.

[73] Technische Universität Dresden. *Grundlagen Hauptkomponentenanalyse*. 2022. URL: https://methpsy.elearning.psych.tu-dresden.de/mediawiki/index.php/Grundlagen_Hauptkomponentenanalyse (besucht am 01. 06. 2024).

[74] Benyamin Bhojogh u. a. *(PDF) Eigenvalue and Generalized Eigenvalue Problems: Tutorial*. 2019. URL: https://arxiv.org/pdf/1903.11240.pdf (besucht am 01. 06. 2024).

[75] Gilbert Tanner. *Linear Discriminant Analysis (LDA)*. 2021. URL: https://ml-explained.com/blog/linear-discriminant-analysis-explained (besucht am 01. 06. 2024).

[76] Priyankur Sarkar. *What is LDA: Linear Discriminant Analysis for Machine Learning*. 2021. URL: https://www.knowledgehut.com/blog/data-science/linear-discriminant-analysis-for-machine-learning (besucht am 01. 06. 2024).

[77] European Commission. *Ethics guidelines for trustworthy AI*. 2019. URL: https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=60425 (besucht am 01. 06. 2024).

[78] Alejandro Arrieta u. a. *Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI*. 2019. URL: http://arxiv.org/pdf/1910.10045.pdf (besucht am 01. 06. 2024).

[79] Zhe Chen u. a. „Interpretable machine learning for building energy management: A state-of-the-art review“. In: *Advances in Applied Energy* 9 (2023), S. 100123. ISSN: 2666-7924. DOI: 10.1016/j.adapen.2023.100123.

[80] Lothar B. Blum. *Angewandte Data Science: Projekte | Methoden | Prozesse*. Springer Fachmedien, 2023. URL: https://link.springer.com/book/10.1007/978-3-658-39625-1.

[81] Bader Aldughayfiq u. a. „Explainable AI for Retinoblastoma Diagnosis: Interpreting Deep Learning Models with LIME and SHAP“. In: *Diagnostics* 13.11 (2023). ISSN: 2075-4418. DOI: 10.3390/diagnostics13111932.

[82] Marco Tulio Ribeiro u. a. *Model-Agnostic Interpretability of Machine Learning*. 2016. DOI: 10.48550/arXiv.1606.05386. (Besucht am 01. 06. 2024).

[83] Rich Caruana u. a. *InterpretML: Explainable Boosting Machines (EBMs)*. 2022. URL: https://people.orie.cornell.edu/mru8/orie4741/lectures/Tutorial4MadeleineUdellClass_2020Dec08_RichCaruana_IntelligibleMLInterpretML_EBMs_75mins.pdf (besucht am 01. 06. 2024).

[84] Harsha Nori u. a. *InterpretML: A Unified Framework for Machine Learning Interpretability*. Sep. 2019. URL: https://arxiv.org/abs/1909.09223 (besucht am 01. 06. 2024).

[85] Yasin Yousif und Jörg Müller. *Efficient and Interpretable Traffic Destination Prediction using Explainable Boosting Machines*. 2024. arXiv: 2402.03457 [cs.LG]. URL: https://arxiv.org/abs/2402.03457.

[86] Indraneel Dutta Baruah. *How Do Inherently Interpretable AI Models Work? — Explainable Boosting Machine*. 2022. URL: https://medium.com/nerd-for-tech/how-do-inherently-interpretable-ai-models-work-explainable-boosting-machine-f9e3718b1a4 (besucht am 01. 06. 2024).

[87] Michael Prof. Dr. Eisermann. *Spieltheorie und ökonomisches Verhalten*. 17. Aug. 2022. URL: https://pnp.mathematik.uni-stuttgart.de/igt/eiserm/lehre/Spieltheorie/Spieltheorie-M-2x2.pdf (besucht am 01. 06. 2024).

[88] Inc C3.ai. *Shapley Values*. 2024. URL: https://c3.ai/glossary/data-science/shapley-values/ (besucht am 01. 06. 2024).

[89] Christoph Molnar. *Interpretable Machine Learning, A Guide for Making Black Box Models Explainable*. 2023. URL: https://christophm.github.io/interpretable-ml-book/ (besucht am 01. 06. 2024).

[90] L. S. Shapley. „17. A Value for n-Person Games“. In: *Contributions to the Theory of Games Vol 2*. Hrsg. von Harold William Kuhn, Albert William Tucker und Albert W. Tucker. Annals of Mathematics Studies. Princeton, NJ: Princeton University Press, 1953, S. 307–318. ISBN: 9781400881970. DOI: 10.1515/9781400881970-018.

[91] C. Molnar. *Interpreting Machine Learning Models with SHAP: A Guide with Python Examples and Theory on Shapley Values*. Chistoph Molnar MUCBOOK, Heidi Seibold, 2023. ISBN: 9798857734445. URL: https://christophmolnar.com/books/shap/.

[92] Scott Lundberg und Lee Su-In. *A Unified Approach to Interpreting Model Predictions*. 2017. URL: https://arxiv.org/abs/1705.07874 (besucht am 01. 06. 2024).

[93] Joran Michiels, Johan Suykens u. a. „Explaining the model and feature dependencies by decomposition of the Shapley value“. In: *Decision Support Systems* 182 (2024), S. 114234. ISSN: 0167-9236. DOI: 10.1016/j.dss.2024.114234.

[94] Dr. Alexander Engelhardt. *Künstliche Intelligenz interpretierbar machen*. 2021. URL: https://www.informatik-aktuell.de/betrieb/kuenstliche-intelligenz/kuenstliche-intelligenz-interpretierbar-machen.html#c30305 (besucht am 01. 06. 2024).

[95] Steven M. Hernandez. *Steven M. Hernandez / ESP32-CSI-Tool*. 2023. URL: https://github.com/StevenMHernandez/ESP32-CSI-Tool/ (besucht am 01. 06. 2024).

[96] Espressif. *Wi-Fi Driver*. 2023. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/wifi.html?highlight=csi (besucht am 01. 06. 2024).

[97] Xiaoqiang Zhu und other. „BLS-Location: A Wireless Fingerprint Localization Algorithm Based on Broad Learning“. In: *IEEE Transactions on Mobile Computing* 22.1 (2023), S. 115–128. URL: https://ieeexplore.ieee.org/abstract/document/9403901.

[98] Zhijia Cai u. a. „Device-Free Wireless Sensing for Gesture Recognition Based on Complementary CSI Amplitude and Phase“. In: (2024). URL: https://www.mdpi.com/1424-8220/24/11/3414 (besucht am 01. 06. 2024).

[99] Bohang Chen u. a. „Research Progress of Wireless Positioning Methods Based on RSSI“. In: (2024). URL: https://www.mdpi.com/2079-9292/13/2/360 (besucht am 01. 06. 2024).

[100] Hicham Boudlal u. a. „A novel approach for simultaneous human activity recognition and pose estimation via skeleton-based leveraging WiFi CSI with YOLOv8 and mediapipe frameworks“. In: (2024). URL: https://link.springer.com/article/10.1007/s11760-024-03031-5 (besucht am 01. 06. 2024).

[101] El Zein Zein u. a. „CSI-based Human Activity Recognition via Lightweight CNN Model and Data Augmentation“. In: (2024). URL: https://ieeexplore.ieee.org/abstract/document/10565773 (besucht am 29. 06. 2024).

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AGC	Automatische Gain Control
AP	Access-Point
ASK	Amplitude-Shift Keying
BPSK	Binary Phase-Shift Keying
CNN	Convolutional Neural Network
CSI	Channel State Information
DA	Data Augmentation
DAC	Digital Analog Converter
DT	Decision Trees (DT)
EBM	Explainable Boosting Machine
FIR-Filter	Finite Impulse Response
FSK	Frequency-Shift Keying
GAM	Generalized Additive Model
GPU	Grafikprozessoreinheit
GRU	Gated Recurrent Units
ICI	Intercarrier Interferenzen
IEEE	Institute of Electrical and Electronics Engineers
IFFT	Inverse Fast-Fourier-Transformation
ISI	Intersymbol Interferenz
KI	Künstliche Intelligenz
LDA	Lineare Diskriminanzanalyse
LLOS	Length Line of Side

MAC	Media-Access-Control Adresse
MIMO	Multiple Input Multiple Output
MISO	Multiple Input Single Output
ML	Maschine-Learning
MLA	Maschine-Learing-Algorithmen
MRC	Maximum Ratio Combining
NB	Naive Bayes
Knn	k-Nearest Neighbors
NRZ	Non-Return-to-Zero-Signals
OFDM	Orthogonal Frequency-Division Multiplexing
PCA	Principal Component Analysis
PDP	Partial Dependence Plots
PFI	Permutation Feature Importance
PSK	Phase-Shift Keying
QAM	Quadratur Amplituden Modulation
QLOS	Quer Line of Side
QPSK	Quadrature Phase-Shift Keying
RR	Round-Robin
RSSI	Received Signal Strength Indicator
SC	Single Carrier
STA	Station
SVM	Support Vector Machine
TX	Transceiver
WLAN	Wireless Local Area Network
WMAS	Weighted Mean Absolute Score
XAI	Explainable Artificial Intelligence

Anhang

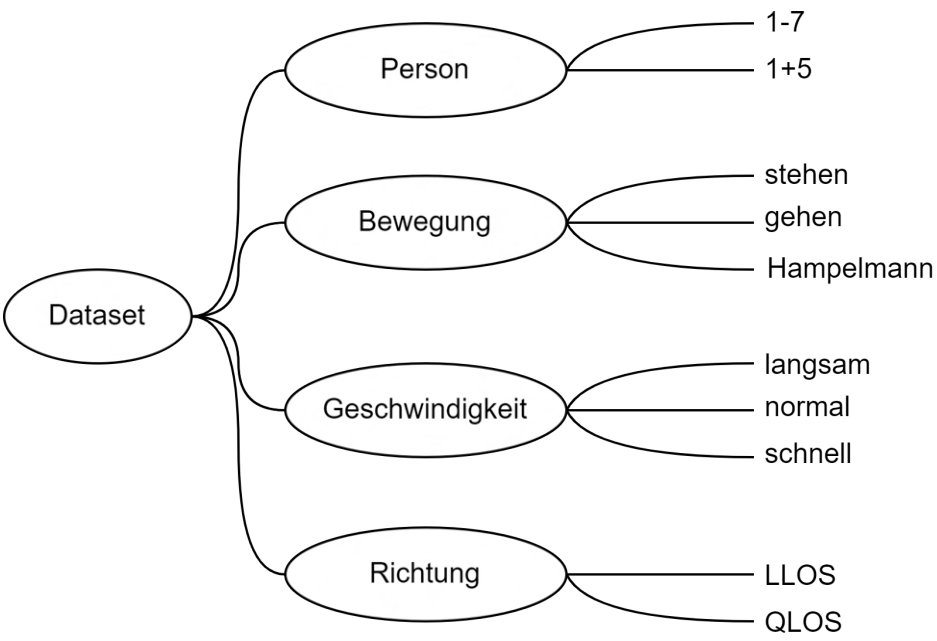


Abbildung 8.1: Mindmap der unterschiedlichen Datensets

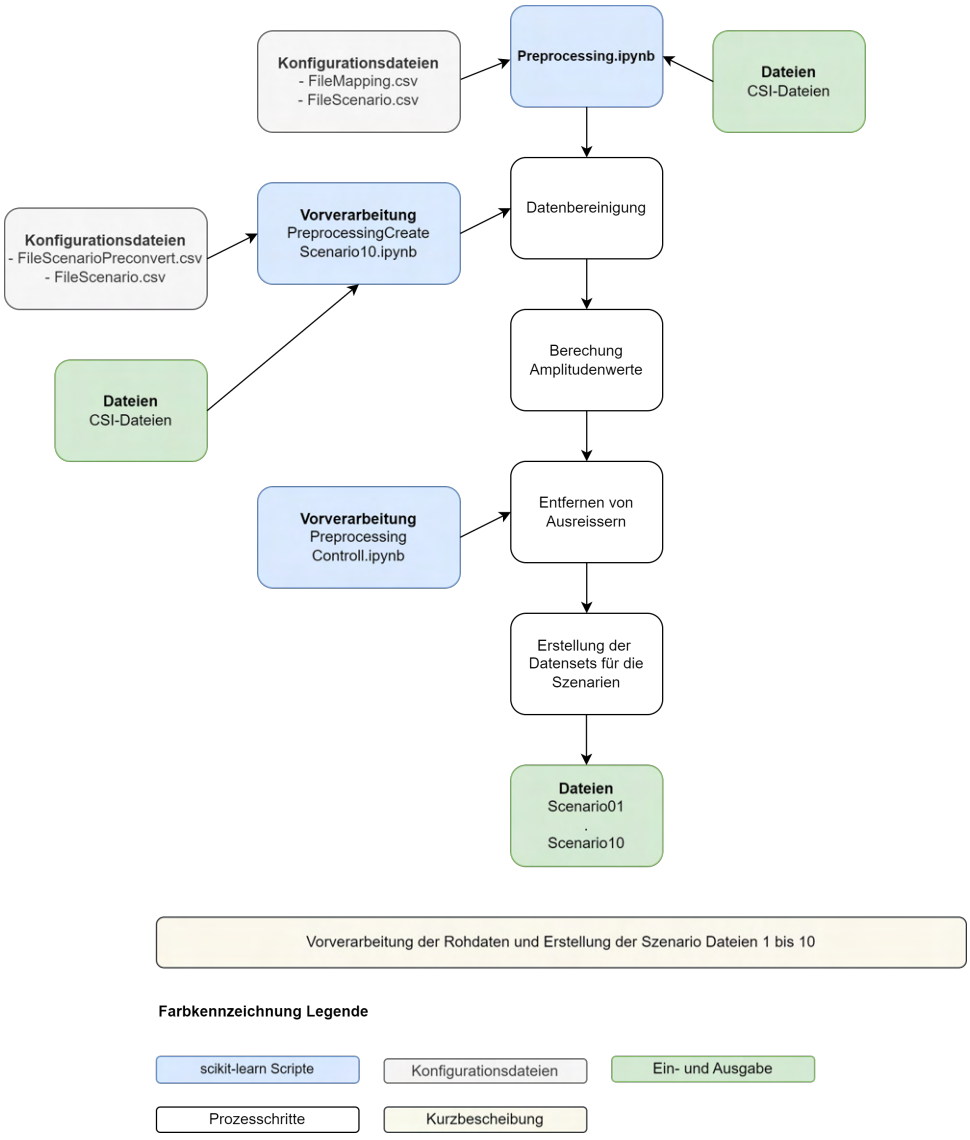


Abbildung 8.2: Ablaufdiagramme der Python Scripte Teil 1 - Vorverarbeitung

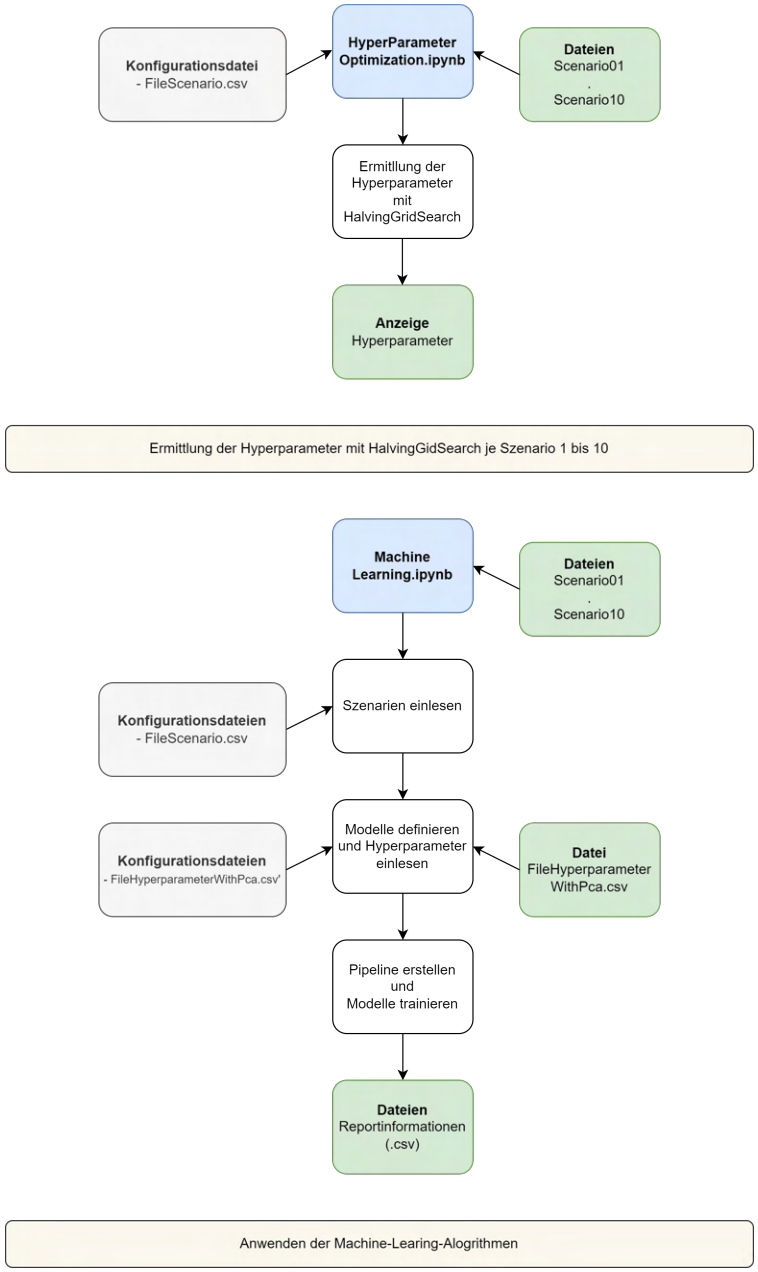


Abbildung 8.3: Ablaufdiagramme der Python Scripte Teil 2 - Hyperparameter und Machine-Learing-Algorithmus

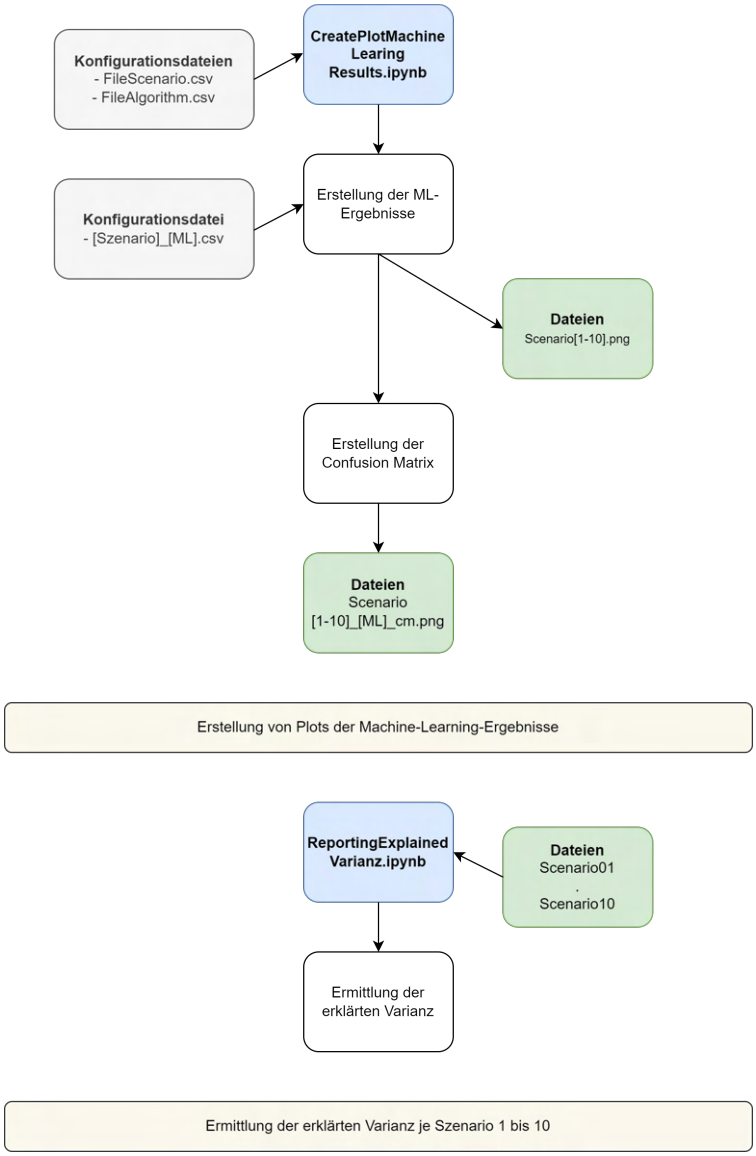


Abbildung 8.4: Ablaufdiagramme der Python Scripte Teil 3 - Generierung Machine-Learning-Ergebnisse und Ermittlung der erklärten Varianz je Szenario 1 bis 10

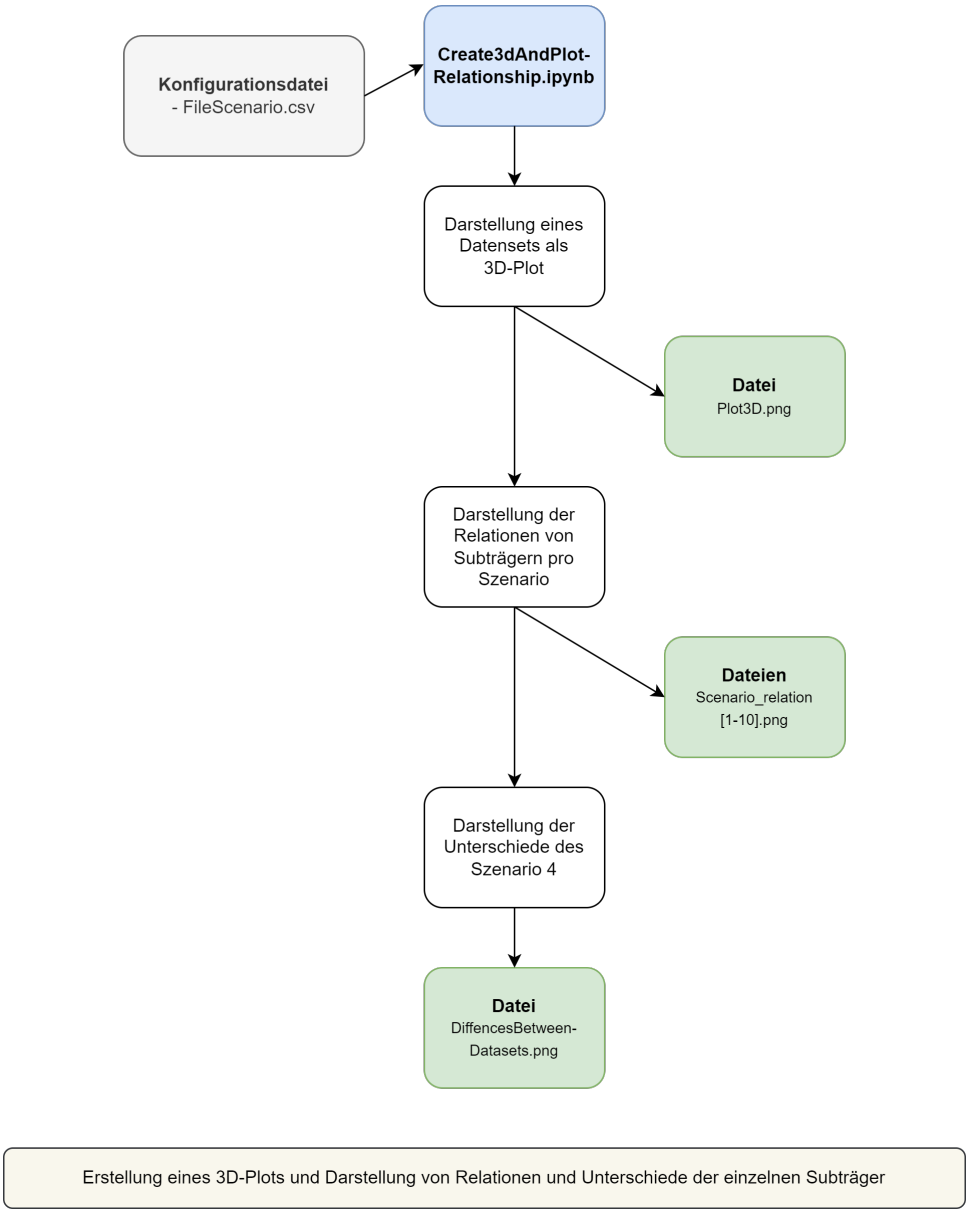


Abbildung 8.5: Ablaufdiagramme der Python Scripte Teil 4 - Erstellung des 3D- und Relationen-Plot

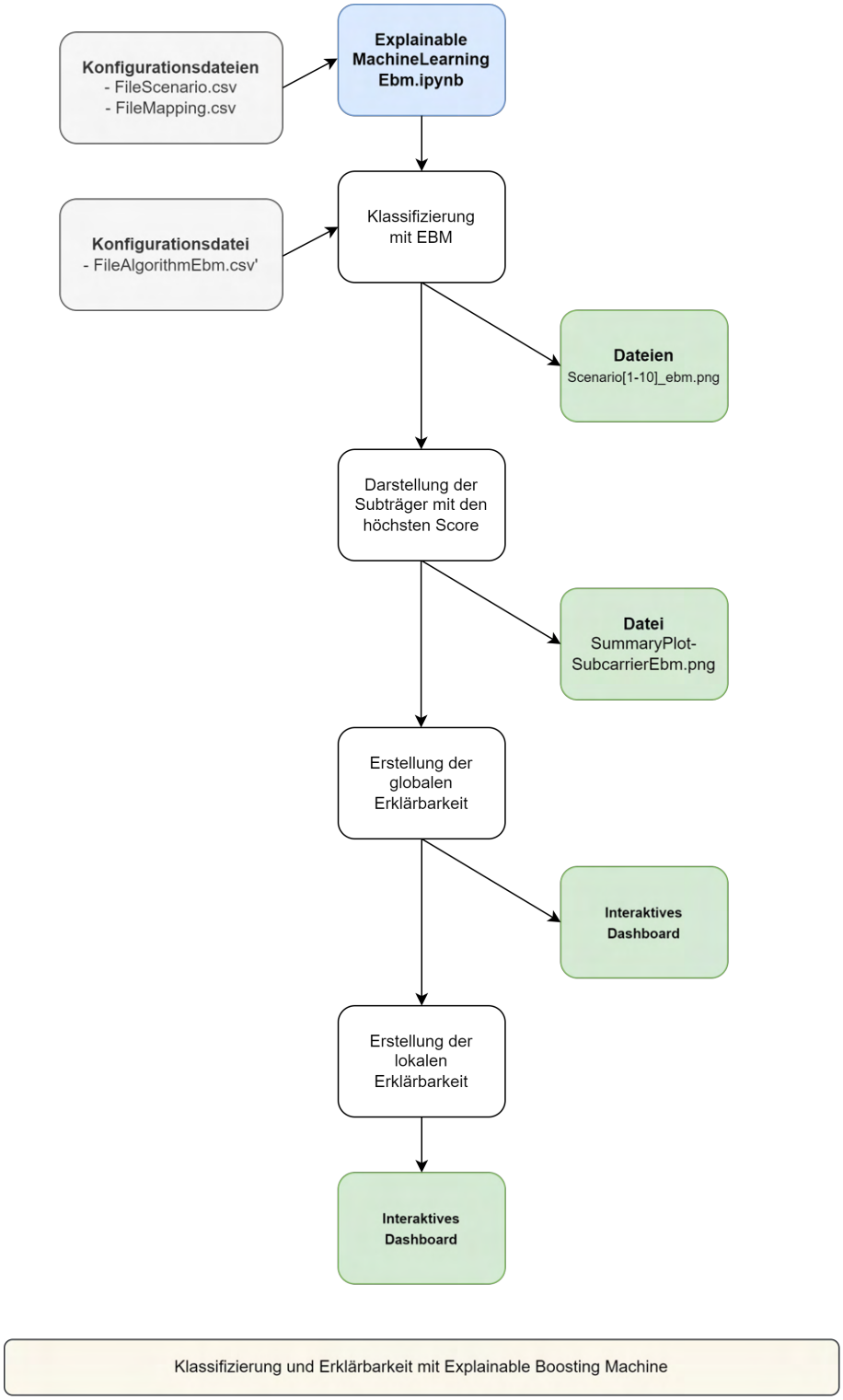


Abbildung 8.6: Ablaufdiagramme der Python Scripte Teil 5 - Explainable Boosting Machine

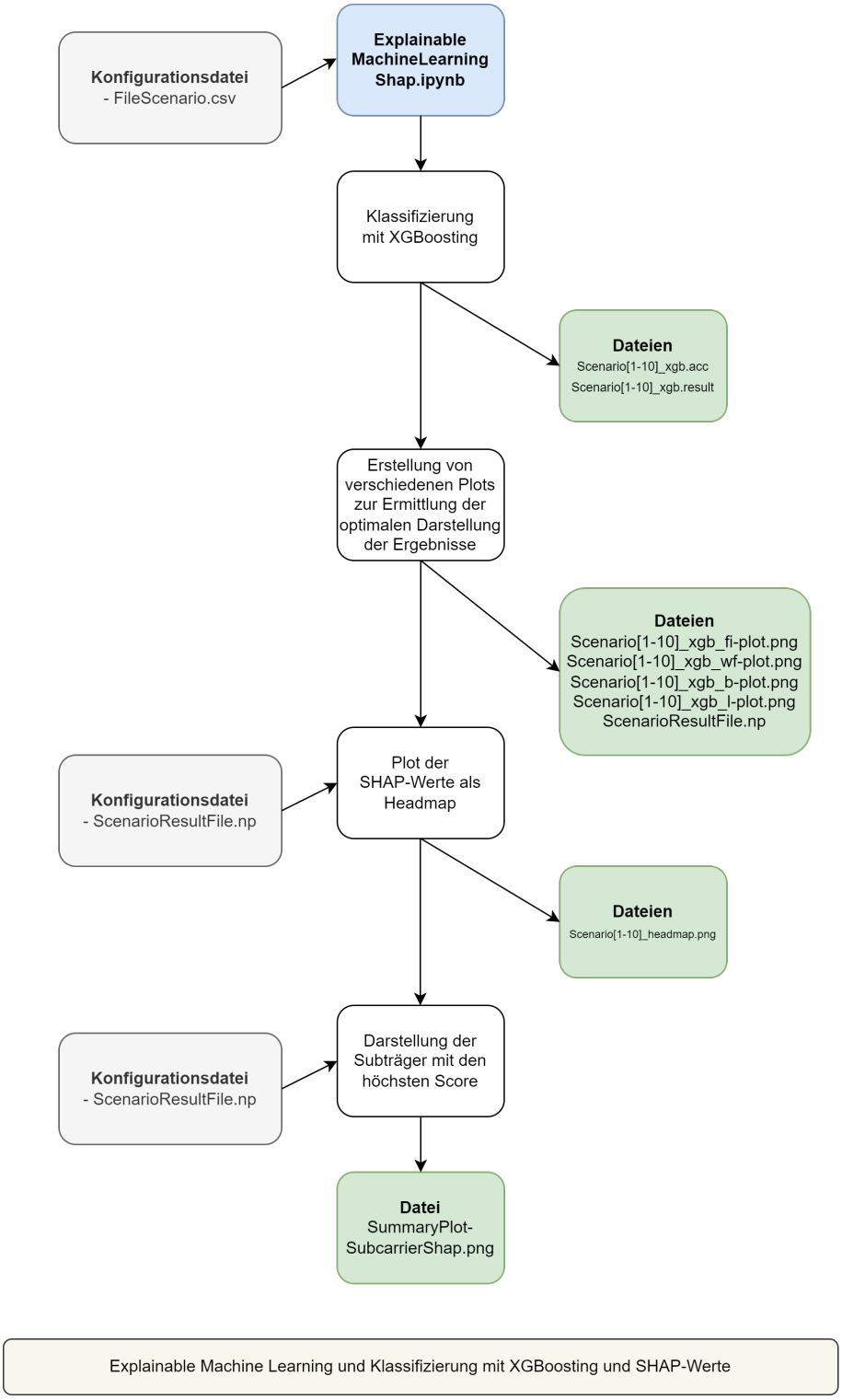


Abbildung 8.7: Ablaufdiagramme der Python Scripte Teil 6 - XGBoost und SHAP

Tabelle 8.1: Übersicht der Machine-Learning-Modelle und Methoden der Erklärbarkeit

	ML Methoden	**Erklärbarkeitsmethoden**			
		SHAP	**LIME**	**Inter-pretML**	**AIX360**
Transparente Modelle (WhiteBox)	Logistic Regression	N	N	Y	Y
	Decision Tree	N	N	Y	Y
	K-Nearest Neighbors	N	N	Y	Y
	Bayesian Model	N	N	Y	Y
	Explainable Boosting Machine	N	N	Y	Y
Nicht transparente Modelle (BlackBox)	Gradient Boosting Model	E	Y	Y	Y
	Support Vector Machines	Y	Y	Y	Y
	Random Forest	Y	Y	Y	Y
	Neuronal Netzworks	E	Y	Y	Y
	Unterstützung durch scikit-learn	T	V	V	T

Legende

N = Eine Kombination dieser Methoden / Modelle ist nicht möglich oder sinnvoll

Y = Eine Kombination dieser Methoden / Modelle ist möglich oder sinnvoll

E = Eine Kombination dieser Methoden / Modelle ist nur eingeschränkt möglich / sinnvoll

V = Vollständige Unterstützung durch scikit-learn

T = Unterstützung durch scikit-learn ist nur teilweise gegeben

Tabelle 8.2: Beschreibung der Kanalzustandsinformationen - Teil 1

Datenfeld	**Bezeichnung**	**Beschreibung**
1	timestamp_received	Timestamp der empfangenen Daten
2	source_mac_addr	MAC-Adresse des Senders
3	csi_recording_id	Eindeutige ID der Aufzeichnung
4	csi_len	Länge der CSI-Daten
5	csi_subcarrier_0	CSI-Daten des Subträgers 0
...	...	...
68	csi_subcarrier_63	CSI-Daten des Subträgers 63
69	rssi	Stärke des empfangenen Signals
70	rate	Übertragungsrate zwischen Sender und Empfänger
71	sig_mode	Signalmodus (z.B. 802.11a/b/g)
72	mcs	Modus der Mehrfachantennennutzung
73	cwb	Kanalbandbreite für die Übertragung
74	aggregation	Paketaggregation (aktiviert / deaktiviert)
75	stbc	Space-Time Block Coding (aktiviert / deaktiviert)
76	fec_coding	Forward Error Correction (aktiviert / deaktiviert)
77	sgi	Short Guard Interval (aktiviert / deaktiviert)
78	noice_floor	Gibt den Pegel des Hintergrundrauschens an
79	ampdu_cnt	Aggregated MAC Protocol Data Unit (dient zur Verbesserung der Datenübertragung)
80	channel	Angabe des aktuellen Funkkanals
81	secondary_channel	Gibt den alternativen Funkkanal an
82	timestamp_device	Zeitpunkt, zu dem die CSI-Daten erfasst wurden

Tabelle 8.3: Beschreibung der Kanalzustandsinformationen - Teil 2

Datenfeld	**Bezeichnung**	**Beschreibung**
83	ant	Informationen zu der physische Konfiguration der Antennen
84	sig_len	Angabe der Anzahl der Abtastwerte
85	rx_state	Receive State (Gibt den Status während des Empfang der Daten an)
86	first_word_invalid	Gibt an, ob der erste Datenblock gültig oder ungültig ist
87	csi_device_id	Eindeutige Identifikationsnummer des Geräts
88	csi_device_name	Name des Devices
89	csi_device_mac	MAC-Adresse des Devices
90	csi_device_ip	IP-Adresse des Devices
91	label	Label der Daten
92	label_id	ID der Daten
93	recording_id	ID der aufgezeichneten Daten
94	name	Name der aufgezeichneten Daten
95	date	Datum der aufgezeichneten Daten
96	length	Länge der aufgezeichneten Daten
97	fps	Rate der CSI-Datenframes pro Sekunde
98	description	Beschreibung aufgezeichneten Daten
99	position	Feld zur Angabe der Position
100	person	Datenfeld zur Angabe der Person
101	temperature	Temperatur des Gerätes
102	categories	Datenfeld zur Angabe der Kategorie

Tabelle 8.4: Zuordnung der Klassen zu den einzelnen Bewegungen - Teil 1

Klasse	**Bewegung**	**Klasse**	**Bewegung**
1	Person-1 Gehen langsam LLOS	26	Person-4 Gehen normal LLOS
2	Person-1 Gehen normal LLOS	27	Person-4 Gehen schnell LLOS
3	Person-1 Gehen schnell LLOS	28	Person-5 Gehen langsam LLOS
4	Person-1 Gehen QLOS #1	29	Person-5 Gehen normal LLOS
5	Person-1 Gehen QLOS #2	30	Person-5 Gehen schnell LLOS
6	Person-1 Gehen QLOS #3	31	Person-5 Gehen QLOS #1
7	Person-1 Hampelfrau QLOS #1	32	Person-5 Gehen QLOS #2
8	Person-1 Stehen QLOS #1	33	Person-5 Gehen QLOS #3
9	Person-1 Hampelfrau LLOS #1	34	Person-5 Hampelmann QLOS #1
10	Person-1 Stehen LLOS #1	35	Person-5 Stehen QLOS #1
11	Person-1 Hampelfrau QLOS #2	36	Person-5 Hampelmann LLOS #1
12	Person-1 Stehen QLOS #2	37	Person-5 Stehen LLOS #1
13	Person-1 Hampelfrau LLOS #2	38	Person-5 Hampelmann QLOS #2
14	Person-1 Stehen LLOS #2	39	Person-5 Stehen QLOS #2
15	Person-1 Hampelfrau QLOS #3	40	Person-5 Hampelmann LLOS #2
16	Person-1 Stehen QLOS #3	41	Person-5 Stehen LLOS #2
17	Person-1 Hampelfrau LLOS #3	42	Person-5 Hampelmann QLOS #3
18	Person-1 Stehen LLOS #3	43	Person-5 Stehen QLOS #3
19	Person-2 Gehen langsam LLOS	44	Person-5 Hampelmann LLOS #3
20	Person-2 Gehen normal LLOS	45	Person-5 Stehen LLOS #3
21	Person-2 Gehen schnell LLOS	46	Person-6 Gehen LLOS langsam
22	Person-3 Gehen langsam LLOS	47	Person-6 Gehen LLOS normal
23	Person-3 Gehen normal LLOS	48	Person-6 Gehen LLOS schnell
24	Person-3 Gehen schnell LLOS	49	Person-6 Gehen quer #1
25	Person-4 Gehen langsam LLOS	50	Person-6 Gehen quer #2

Tabelle 8.5: Zuordnung der Klassen zu den einzelnen Bewegungen - Teil 2

Klasse	**Bewegung**
51	Person-6 Gehen quer #3
52	Person-6 Hampelmann quer #1
53	Person-6 Stehen quer #1
54	Person-6 Hampelmann längs #1
55	Person-6 Stehen längs #1
56	Person-6 Hampelmann quer #2
57	Person-6 Stehen quer #2
58	Person-6 Hampelmann längs #2
59	Person-6 Stehen längs #2
60	Person-6 Hampelmann quer #3
61	Person-6 Stehen quer #3
62	Person-6 Hampelmann längs #3
63	Person-6 Stehen längs #3
64	Person-7 Gehen langsam LLOS
65	Person-7 Gehen normal LLOS
66	Person-7 Gehen schnell LLOS
67	Personen-8 Gehen langsam LLOS
68	Personen-8 Gehen normal LLOS
69	Personen-8 Gehen schnell LLOS
70	Raum leer
71	Person-5 AlleBewegungen #1
72	Person-5 AlleBewegungen #2
73	Person-5 AlleBewegungen #3

Tabelle 8.6: Personenidentifikation - Zuordnung der Bewegungen aller Personen zu den Szenarien

Szenario	**Bewegung**	**Person ID**	**Samples**
1	Gehen langsam von allen Personen	1-8	371.150
2	Gehen normal von allen Personen	1-8	374.083
3	Gehen schnell von allen Personen	1-8	370.736

Tabelle 8.7: Bewegungsdetektion - Zuordnung der Bewegung der Person 5 zu den Szenarien

Szenario	**Bewegung**	**Person ID**	**Samples**
4	Gehen langsam, normal, schnell	5	144.197
5	Alle Bewegungen der Person außer gehen langsam, normal, schnell	5	704.932
6	Alle Bewegungen der Person	5	849.128

Tabelle 8.8: Positionsdetektion - Zuordnung von Bewegungen in Abhängig von der Position

Szenario	**Bewegung**	**Person ID**	**Samples**
7	Gehen quer, Hampelmann quer, stehen quer, Hampelmann längs, stehen längs der Person an Position #1	5	236.284
8	Gehen quer, Hampelmann quer, stehen quer, Hampelmann längs, stehen längs der Person an Position #2	5	238.245
9	Gehen quer, Hampelmann quer, stehen quer, Hampelmann längs, stehen längs der Person an Position #3	5	230.405
10	Alle Bewegungen einer Person an Position #1, #2, #3	5	704.922

Tabelle 8.9: Zuordnung der Datensets zu den Szenarien

Szenario	**Datensets**
1	1, 19, 22, 25, 28, 46,64, 67
2	2, 20,23, 26, 29, 47, 65, 68
3	3, 21, 24, 27, 39, 48, 66, 69
4	28, 29, 30
5	37, 41, 45, 35, 39, 43, 31, 32, 33, 36, 40, 44, 34, 38, 42
6	37, 41, 45, 35, 39, 43, 31, 32, 33, 28, 29, 30, 36, 40, 44, 34, 38, 42
7	37, 35, 31, 36, 34
8	32, 38, 39, 40, 41
9	45, 43, 33, 44, 42
10	71, 72, 73

Tabelle 8.10: Zuordnungen der bestehenden Datensets zu den kummulierten Datensets 71, 72 und 73

Datensets	**Kummuliertes Datenset**	**Beschreibung**
31, 34, 35, 36, 37	71	Alle Bewegungen der Person 5 an Position #1
32, 38, 39, 40, 41	72	Alle Bewegungen der Person 5 an Position #2
33, 42, 43, 44, 45	73	Alle Bewegungen der Person 5 an Position #3

Tabelle 8.11: Erklärte Varianz der Szenarien 1-10

		Szenario									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Komponent:	1	0,463	0,457	0,464	0,461	0,542	0,526	0,500	0,561	0,509	0,548
Komponent:	2	0,640	0,648	0,653	0,640	0,697	0,686	0,682	0,702	0,676	0,701
Komponent:	3	0,769	0,774	0,779	0,775	0,793	0,788	0,789	0,769	0,790	0,797
Komponent:	4	0,819	0,828	0,833	0,841	0,837	0,838	0,831	0,809	0,842	0,837
Komponent:	5	0,851	0,858	0,858	0,867	0,856	0,858	0,849	0,837	0,861	0,857
Komponent:	6	0,872	0,877	0,877	0,887	0,870	0,874	0,862	0,850	0,871	0,870
Komponent:	7	0,881	0,885	0,885	0,894	0,877	0,880	0,870	0,858	0,878	0,878
Komponent:	8	0,886	0,892	0,891	0,901	0,883	0,886	0,876	0,864	0,884	0,884
Komponent:	9	0,891	0,896	0,896	0,905	0,888	0,891	0,881	0,870	0,888	0,889
Komponent:	10	0,896	0,901	0,900	0,909	0,892	0,896	0,886	0,876	0,893	0,893
Komponent:	11	0,900	0,904	0,904	0,912	0,896	0,900	0,890	0,881	0,897	0,897
Komponent:	12	0,903	0,908	0,907	0,915	0,900	0,903	0,894	0,885	0,900	0,901
Komponent:	13	0,906	0,911	0,910	0,918	0,903	0,906	0,897	0,889	0,903	0,904
Komponent:	14	0,909	0,914	0,913	0,921	0,906	0,909	0,901	0,893	0,906	0,907
Komponent:	15	0,912	0,916	0,916	0,924	0,909	0,912	0,904	0,897	0,909	0,910
Komponent:	16	0,915	0,919	0,918	0,926	0,912	0,915	0,907	0,901	0,912	0,913
Komponent:	17	0,918	0,922	0,921	0,928	0,915	0,918	0,910	0,905	0,915	0,916
Komponent:	18	0,920	0,924	0,924	0,931	0,918	0,920	0,913	0,909	0,918	0,919
Komponent:	19	0,923	0,927	0,927	0,933	0,920	0,923	0,916	0,913	0,921	0,921
Komponent:	20	0,926	0,930	0,929	0,935	0,923	0,926	0,919	0,916	0,923	0,924
Komponent:	21	0,928	0,932	0,932	0,938	0,926	0,928	0,921	0,920	0,926	0,927
Komponent:	22	0,931	0,935	0,934	0,940	0,928	0,931	0,924	0,923	0,929	0,929
Komponent:	23	0,933	0,937	0,937	0,942	0,931	0,933	0,927	0,926	0,931	0,932
Komponent:	24	0,936	0,940	0,939	0,944	0,934	0,936	0,930	0,929	0,934	0,935
Komponent:	25	0,939	0,942	0,942	0,947	0,936	0,938	0,933	0,932	0,937	0,938
Komponent:	26	0,941	0,945	0,944	0,949	0,939	0,941	0,935	0,935	0,939	0,940
Komponent:	27	0,944	0,947	0,946	0,951	0,941	0,943	0,938	0,938	0,942	0,943
Komponent:	28	0,946	0,949	0,949	0,953	0,944	0,946	0,941	0,941	0,944	0,945
Komponent:	29	0,949	0,952	0,951	0,955	0,946	0,948	0,943	0,944	0,947	0,948
Komponent:	30	0,951	0,954	0,953	0,957	0,949	0,950	0,946	0,946	0,949	0,950
Komponent:	31	0,954	0,956	0,956	0,959	0,952	0,953	0,949	0,949	0,952	0,953
Komponent:	32	0,956	0,959	0,958	0,962	0,954	0,956	0,951	0,952	0,954	0,955
Komponent:	33	0,958	0,961	0,960	0,964	0,956	0,958	0,954	0,955	0,957	0,958
Komponent:	34	0,961	0,963	0,963	0,966	0,959	0,960	0,957	0,957	0,959	0,960
Komponent:	35	0,963	0,966	0,965	0,968	0,961	0,963	0,959	0,960	0,962	0,963
Komponent:	36	0,965	0,968	0,967	0,970	0,964	0,965	0,962	0,963	0,964	0,965
Komponent:	37	0,968	0,970	0,969	0,972	0,966	0,967	0,965	0,965	0,966	0,968
Komponent:	38	0,970	0,972	0,971	0,974	0,969	0,970	0,967	0,968	0,969	0,970
Komponent:	39	0,973	0,974	0,974	0,976	0,971	0,972	0,970	0,970	0,971	0,972
Komponent:	40	0,975	0,976	0,976	0,978	0,974	0,974	0,972	0,973	0,974	0,975
Komponent:	41	0,977	0,979	0,978	0,980	0,976	0,977	0,975	0,975	0,976	0,977
Komponent:	42	0,979	0,981	0,980	0,982	0,978	0,979	0,977	0,978	0,978	0,979
Komponent:	43	0,982	0,983	0,982	0,984	0,981	0,981	0,980	0,980	0,981	0,981
Komponent:	44	0,984	0,985	0,984	0,986	0,983	0,983	0,982	0,983	0,983	0,984
Komponent:	45	0,986	0,987	0,986	0,988	0,985	0,986	0,984	0,985	0,985	0,986
Komponent:	46	0,988	0,989	0,988	0,990	0,987	0,988	0,987	0,987	0,987	0,988
Komponent:	47	0,990	0,991	0,990	0,991	0,989	0,990	0,989	0,989	0,989	0,990
Komponent:	48	0,992	0,993	0,992	0,993	0,992	0,992	0,991	0,992	0,992	0,992
Komponent:	49	0,994	0,994	0,994	0,995	0,994	0,994	0,993	0,994	0,994	0,994
Komponent:	50	0,996	0,996	0,996	0,997	0,996	0,996	0,996	0,996	0,996	0,996
Komponent:	51	0,998	0,998	0,998	0,998	0,998	0,998	0,998	0,998	0,998	0,998
Komponent:	52	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000

Tabelle 8.12: Liste der ermittelten Hyperparameter - Teil 1

	Szenario			
Hyperparameter	**1**	**2**	**3**	**4**
RF___max_depth	6	6	6	6
RF___min_samples_leaf	5	3	2	6
RF___min_samples_split	3	5	4	3
RF___n_estimators	200	250	250	250
LR___C	1.0	1.0	0.5	0.5
LR___max_iter	2500	2500	2500	2500
LR___penalty	l2	l2	l2	l2
LR___solver	liblinear	liblinear	liblinear	liblinear
KNN___metric	manhattan	manhattan	manhattan	manhattan
KNN___n_neighbors	6	6	6	6
KNN___weights	distance	distance	distance	distance
SVM___C	6	6	6	6
SVM___gamma	auto	auto	auto	auto
SVM___kernel	rbf	rbf	rbf	rbf
GBC___max_depth	6	6	6	6
GBC___min_samples_leaf	4	5	4	5
GBC___min_samples_split	6	5	6	2

Tabelle 8.13: Liste der ermittelten Hyperparameter - Teil 2

	Szenario		
Hyperparameter	**5**	**6**	**7**
RF___max_depth	6	6	6
RF___min_samples_leaf	2	2	1
RF___min_samples_split	3	3	5
RF___n_estimators	250	200	250
LR___C	0.5	1.0	0.5
LR___max_iter	2500	2500	2500
LR___penalty	l2	l2	l2
LR___solver	liblinear	liblinear	liblinear
KNN___metric	manhattan	manhattan	manhattan
KNN___n_neighbors	6	6	4
KNN___weights	distance	distance	distance
SVM___C	6	6	6
SVM___gamma	auto	auto	auto
SVM___kernel	rbf	rbf	rbf
GBC___max_depth	6	6	6
GBC___min_samples_leaf	6	6	4
GBC___min_samples_split	5	3	5

Tabelle 8.14: Liste der ermittelten Hyperparameter - Teil 3

	Szenario		
Hyperparameter	**8**	**9**	**10**
RF___max_depth	6	6	6
RF___min_samples_leaf	2	1	4
RF___min_samples_split	5	2	6
RF___n_estimators	150	200	200
LR___C	1.0	0.1	1.0
LR___max_iter	2500	2500	2500
LR___penalty	l1	l2	l1
LR___solver	liblinear	liblinear	liblinear
KNN___metric	manhattan	manhattan	manhattan
KNN___n_neighbors	4	6	6
KNN___weights	distance	distance	distance
SVM___C	6	6	6
SVM___gamma	auto	auto	auto
SVM___kernel	rbf	rbf	rbf
GBC___max_depth	6	6	6
GBC___min_samples_leaf	2	5	4
GBC___min_samples_split	6	4	5

Tabelle 8.15: Hyperparameter Teil 3

Tabelle 8.16: Explainable Boosting Machine - Vergleich der Ergebnisse mit und ohne Hyperparameteroptimierung

Ohne Hyperparameter-optimiertung		**Mit Hyperparameter-optimiertung**		**Differenz in Prozent**
Minuten	**F1-Score**	**Minuten**	**F1-Score**	**F1-Score**
53,43	0,4646	20,65	0,3686	9,60
53,91	0,4572	19,92	0,3563	10,09
38,32	0,3435	17,77	0,3433	0,02
1,23	0,5105	0,78	0,5111	0,06
230,53	0,6801	112,13	0,6800	0,01
339,37	0,5717	168,12	0,5717	0,00
11,95	0,7671	6,37	0,8267	5,96
11,09	0,8614	6,90	0,8009	6,05
13,41	0,8148	6,04	0,8153	0,05
239,02	0,7741	113,62	0,7740	0,01

Tabelle 8.17: Explainable Boosting Machine - Hyperparameter

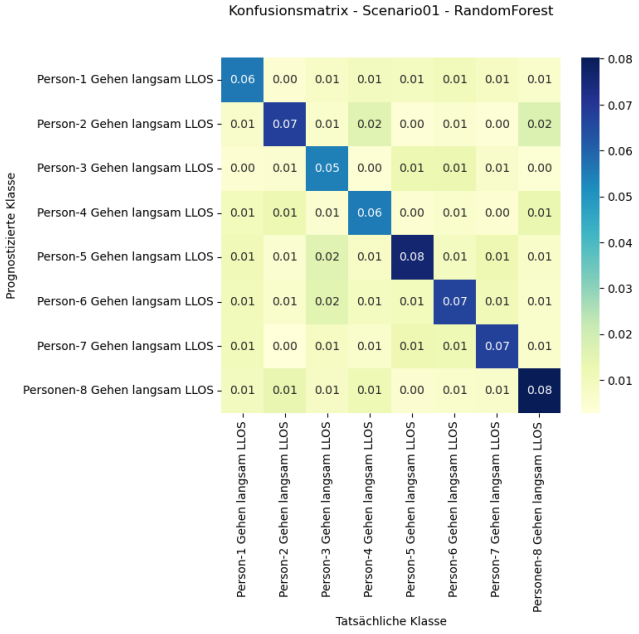
	Szenario									
Hyperparameter	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
learning_rate	0.02	0.02	0.02	0.01	0.02	0.02	0.02	0.02	0.02	0.02
max_bins	256	512	1024	512	1024	256	512	512	512	256
max_interaction_bins	64	32	16	32	32	16	32	32	32	32
max_rounds	5000	5000	5000	5000	5000	5000	5000	5000	5000	5000
min_samples_leaf	3	3	3	3	3	2	3	3	3	2
outer_bags	50	50	75	14	25	50	50	100	25	50
validation_size	0.1	0.1	0.15	0.1	0.1	0.1	0.1	0.1	0.15	0.1

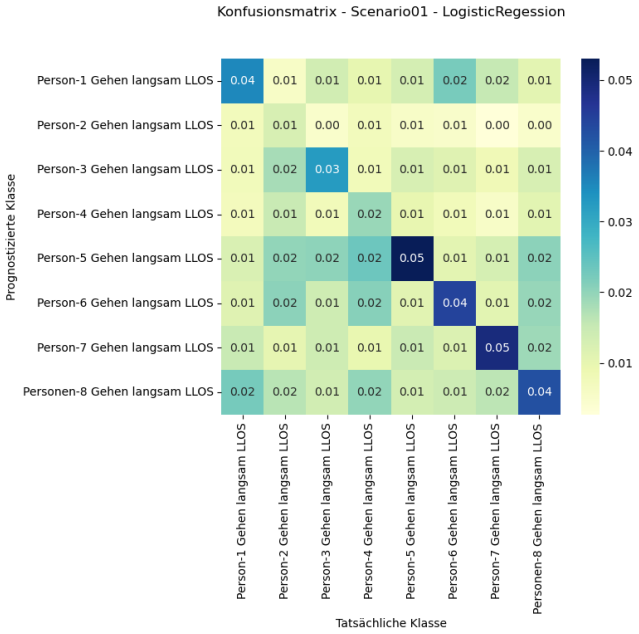
Tabelle 8.18: XGBoost - Vergleich der Ergebnisse mit und ohne Hyperparameteroptimierung

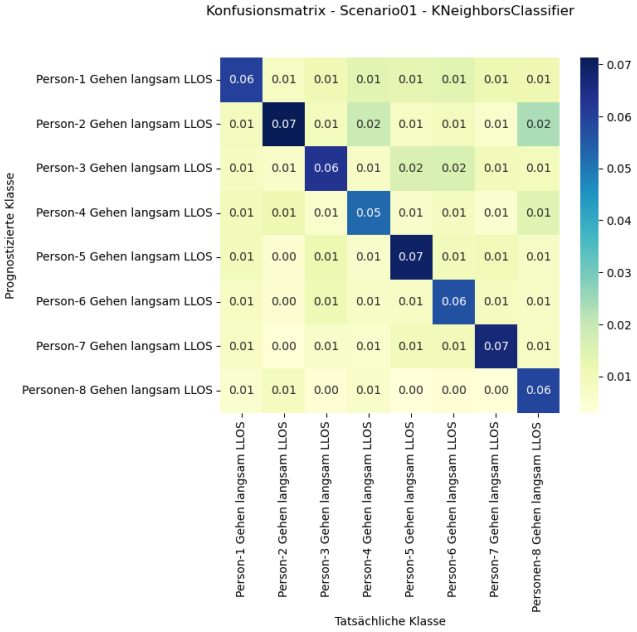
Ohne Hyperparameteroptimiertung		**Mit Hyperparameteroptimiertung**		**Differenz in Prozent**
Sekunden	**F1-Score**	**Sekunden**	**F1-Score**	**F1-Score**
7	0,50	12	0,51	1,0
6	0,47	17	0,53	6,0
6	0,44	18	0,48	4,0
2	0,57	2	0,58	1,0
18	0,80	50	0,83	3,0
7	0,69	70	0,72	3,0
3	0,92	8	0,94	2,0
3	0,90	7	0,91	1,0
3	0,91	8	0,92	1,0
7	0,87	13	0,90	3,0

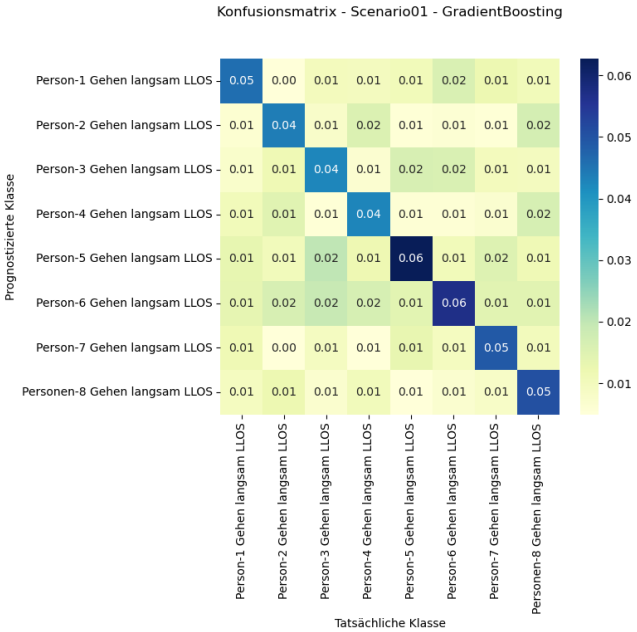
Tabelle 8.19: XGBoost - Hyperparameter

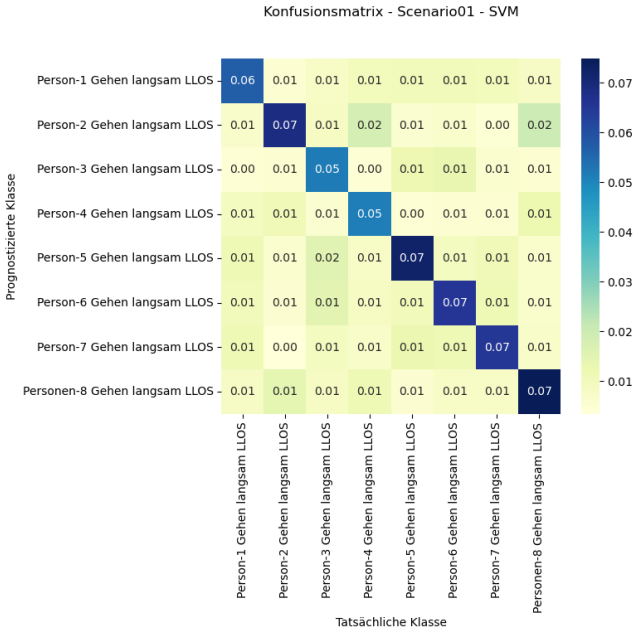
	Szenario									
Hyperparameter	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
colsample_bytree	0.7	0.5	1.0	0.5	0.5	0.5	0.5	0.7	0.7	0.7
eta	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1	0.1	0.3
gamma	0.3	0.3	0	0	0.1	0.1	0	0	0	0
max_depth	7	9	9	9	9	9	9	9	9	9
min_child_weight	1	3	1	5	1	1	1	1	1	1
n_estimators	200	200	200	100	200	200	200	200	200	200
subsample	0.5	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	1.0

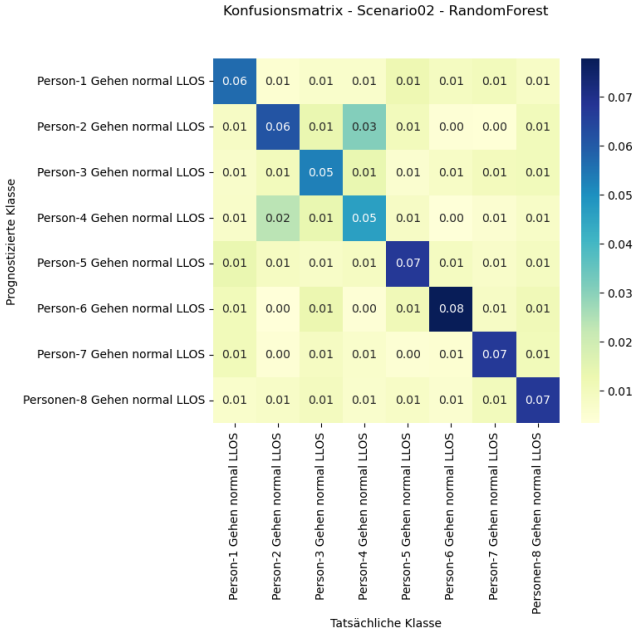

Konfusionsmatrix - Scenario01 - RandomForest
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen langsam LLOS
Person-2 Gehen langsam LLOS
Person-3 Gehen langsam LLOS
Person-4 Gehen langsam LLOS
Person-5 Gehen langsam LLOS
Person-6 Gehen langsam LLOS
Person-7 Gehen langsam LLOS
Personen-8 Gehen langsam LLOS
0.06 0.00 0.01 0.01 0.01 0.01 0.01 0.01
0.01 0.07 0.01 0.02 0.00 0.01 0.00 0.02
0.00 0.01 0.05 0.00 0.01 0.01 0.01 0.00
0.01 0.01 0.01 0.06 0.00 0.01 0.00 0.01
0.01 0.01 0.02 0.01 0.08 0.01 0.01 0.01
0.01 0.01 0.02 0.01 0.01 0.07 0.01 0.01
0.01 0.00 0.01 0.01 0.01 0.01 0.07 0.01
0.01 0.01 0.01 0.01 0.00 0.01 0.01 0.08
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01

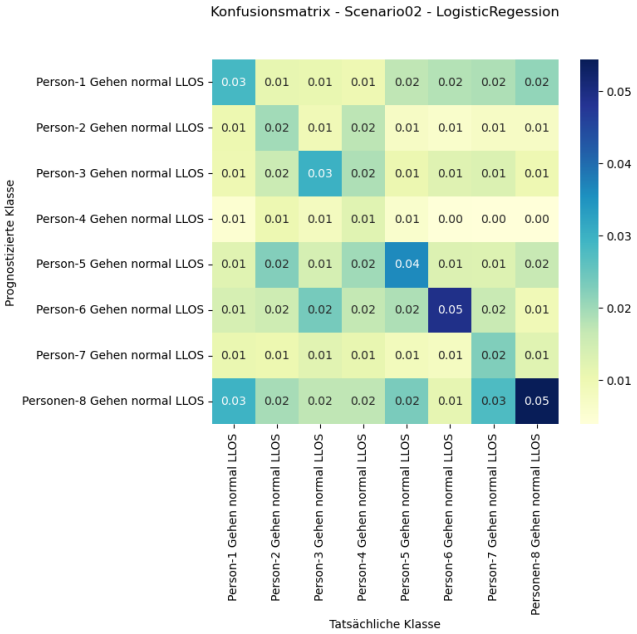

Konfusionsmatrix - Scenario01 - LogisticRegession
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen langsam LLOS
Person-2 Gehen langsam LLOS
Person-3 Gehen langsam LLOS
Person-4 Gehen langsam LLOS
Person-5 Gehen langsam LLOS
Person-6 Gehen langsam LLOS
Person-7 Gehen langsam LLOS
Personen-8 Gehen langsam LLOS
0.04 0.01 0.01 0.01 0.01 0.02 0.02 0.01
0.01 0.01 0.00 0.01 0.01 0.01 0.00 0.00
0.01 0.02 0.03 0.01 0.01 0.01 0.01 0.01
0.01 0.01 0.01 0.02 0.01 0.01 0.01 0.01
0.01 0.02 0.02 0.02 0.05 0.01 0.01 0.02
0.01 0.02 0.01 0.02 0.01 0.04 0.01 0.02
0.01 0.01 0.01 0.01 0.01 0.01 0.05 0.02
0.02 0.02 0.01 0.02 0.01 0.01 0.02 0.04
0.05
0.04
0.03
0.02
0.01

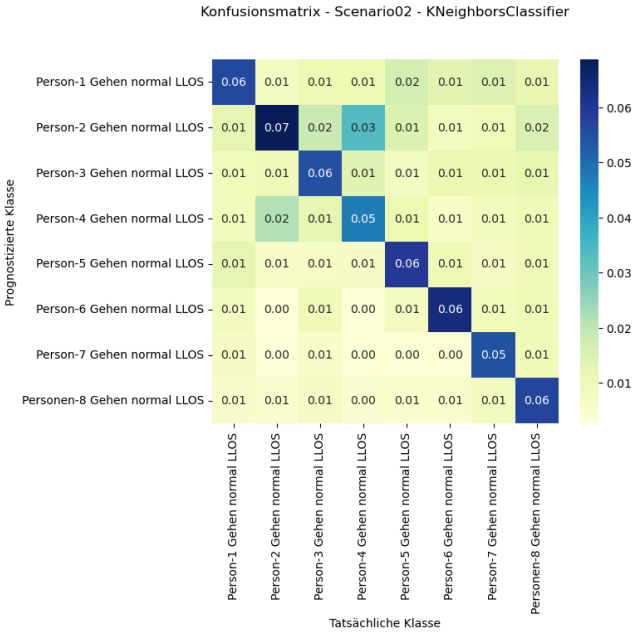

Konfusionsmatrix - Scenario01 - KNeighborsClassifier
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen langsam LLOS
Person-2 Gehen langsam LLOS
Person-3 Gehen langsam LLOS
Person-4 Gehen langsam LLOS
Person-5 Gehen langsam LLOS
Person-6 Gehen langsam LLOS
Person-7 Gehen langsam LLOS
Personen-8 Gehen langsam LLOS
0.06 0.01 0.01 0.01 0.01 0.01 0.01 0.01
0.01 0.07 0.01 0.02 0.01 0.01 0.01 0.02
0.01 0.01 0.06 0.01 0.02 0.02 0.01 0.01
0.01 0.01 0.01 0.05 0.01 0.01 0.01 0.01
0.01 0.00 0.01 0.01 0.07 0.01 0.01 0.01
0.01 0.00 0.01 0.01 0.01 0.06 0.01 0.01
0.01 0.00 0.01 0.01 0.01 0.01 0.07 0.01
0.01 0.01 0.00 0.01 0.00 0.00 0.00 0.06
0.07
0.06
0.05
0.04
0.03
0.02
0.01

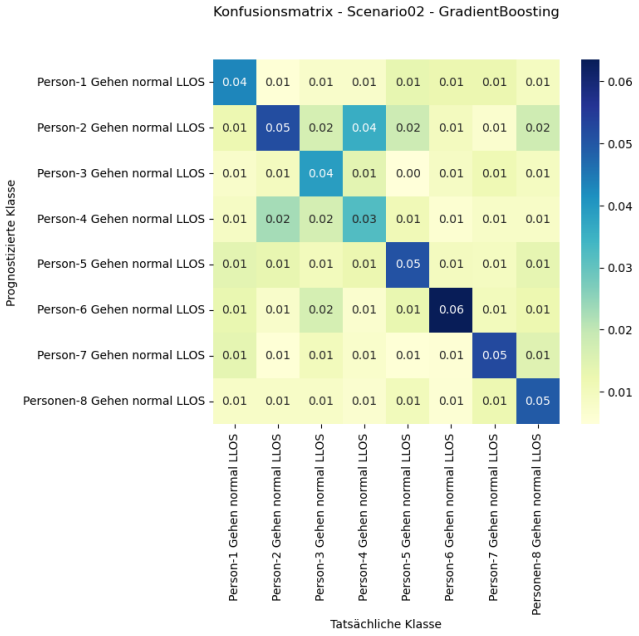

Konfusionsmatrix - Scenario01 - GradientBoosting
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen langsam LLOS
Person-2 Gehen langsam LLOS
Person-3 Gehen langsam LLOS
Person-4 Gehen langsam LLOS
Person-5 Gehen langsam LLOS
Person-6 Gehen langsam LLOS
Person-7 Gehen langsam LLOS
Personen-8 Gehen langsam LLOS
0.05 0.00 0.01 0.01 0.01 0.02 0.01 0.01
0.01 0.04 0.01 0.02 0.01 0.01 0.01 0.02
0.01 0.01 0.04 0.01 0.02 0.02 0.01 0.01
0.01 0.01 0.01 0.04 0.01 0.01 0.01 0.02
0.01 0.01 0.02 0.01 0.06 0.01 0.02 0.01
0.01 0.02 0.02 0.02 0.01 0.06 0.01 0.01
0.01 0.00 0.01 0.01 0.01 0.01 0.05 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.05
0.06
0.05
0.04
0.03
0.02
0.01

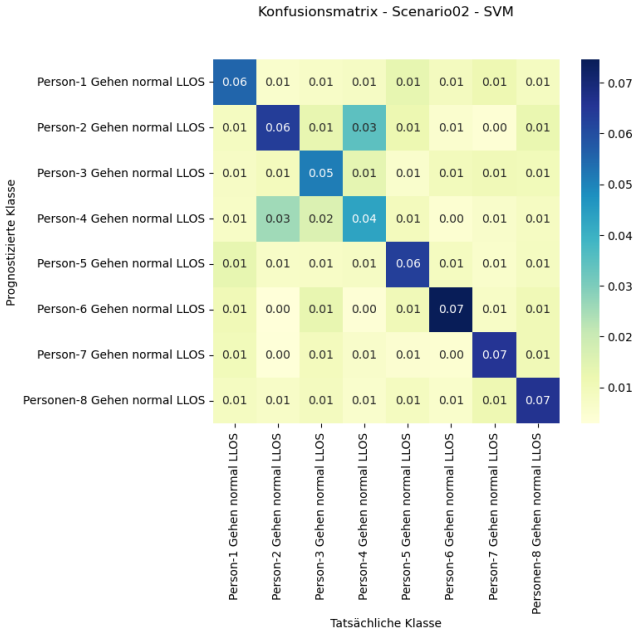

Konfusionsmatrix - Scenario01 - SVM
Prognostizierte Klasse
Person-1 Gehen langsam LLOS
Person-2 Gehen langsam LLOS
Person-3 Gehen langsam LLOS
Person-4 Gehen langsam LLOS
Person-5 Gehen langsam LLOS
Person-6 Gehen langsam LLOS
Person-7 Gehen langsam LLOS
Personen-8 Gehen langsam LLOS
Tatsächliche Klasse
0.07
0.06
0.05
0.04
0.03
0.02
0.01

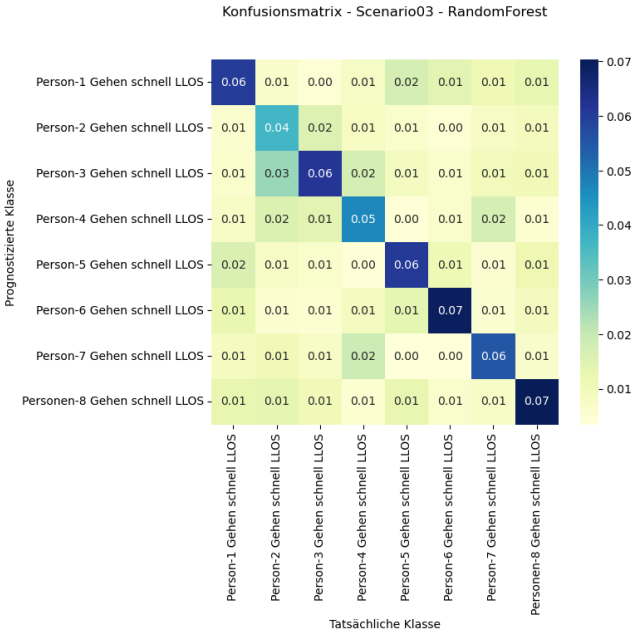

Konfusionsmatrix - Scenario02 - RandomForest
Prognostizierte Klasse
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
Tatsächliche Klasse
0.07
0.06
0.05
0.04
0.03
0.02
0.01

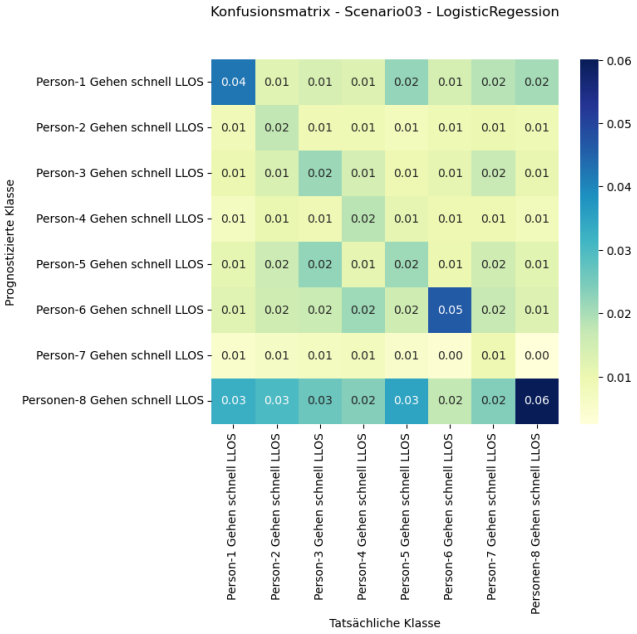

Konfusionsmatrix - Scenario02 - LogisticRegession
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
0.03 0.01 0.01 0.01 0.02 0.02 0.02 0.02
0.01 0.02 0.01 0.02 0.01 0.01 0.01 0.01
0.01 0.02 0.03 0.02 0.01 0.01 0.01 0.01
0.01 0.01 0.01 0.01 0.01 0.00 0.00 0.00
0.01 0.02 0.01 0.02 0.04 0.01 0.01 0.02
0.01 0.02 0.02 0.02 0.02 0.05 0.02 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.02 0.01
0.03 0.02 0.02 0.02 0.02 0.01 0.03 0.05
0.05
0.04
0.03
0.02
0.01

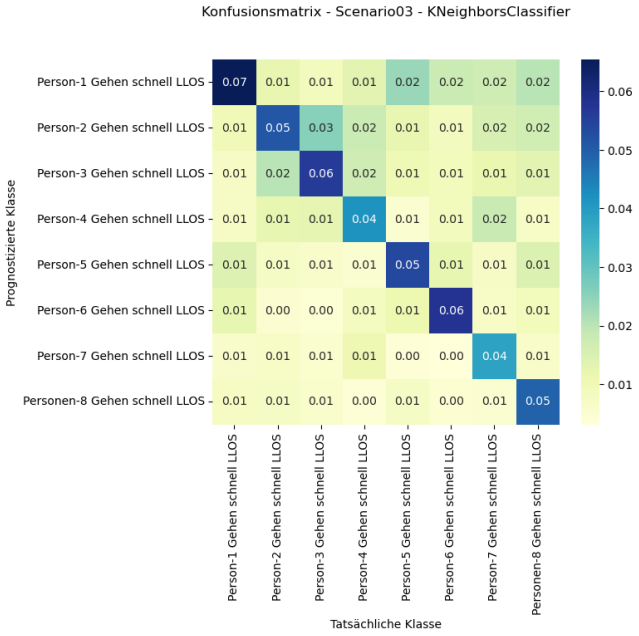

Konfusionsmatrix - Scenario02 - KNeighborsClassifier
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
0.06 0.01 0.01 0.01 0.02 0.01 0.01 0.01
0.01 0.07 0.02 0.03 0.01 0.01 0.01 0.02
0.01 0.01 0.06 0.01 0.01 0.01 0.01 0.01
0.01 0.02 0.01 0.05 0.01 0.01 0.01 0.01
0.01 0.01 0.01 0.01 0.06 0.01 0.01 0.01
0.01 0.00 0.01 0.00 0.01 0.06 0.01 0.01
0.01 0.00 0.01 0.00 0.00 0.00 0.05 0.01
0.01 0.01 0.01 0.00 0.01 0.01 0.01 0.06
0.06
0.05
0.04
0.03
0.02
0.01

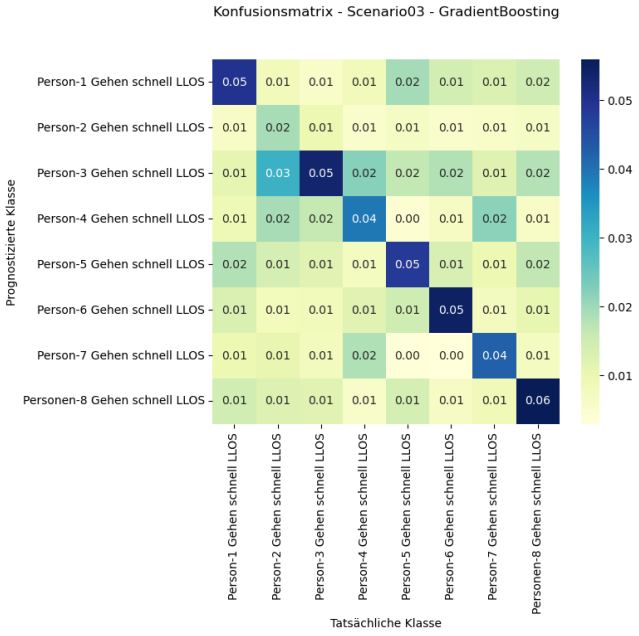

Konfusionsmatrix - Scenario02 - GradientBoosting
Prognostizierte Klasse
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
0.04 0.01 0.01 0.01 0.01 0.01 0.01 0.01
0.01 0.05 0.02 0.04 0.02 0.01 0.01 0.02
0.01 0.01 0.04 0.01 0.00 0.01 0.01 0.01
0.01 0.02 0.02 0.03 0.01 0.01 0.01 0.01
0.01 0.01 0.01 0.01 0.05 0.01 0.01 0.01
0.01 0.01 0.02 0.01 0.01 0.06 0.01 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.05 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.05
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
Tatsächliche Klasse
0.06
0.05
0.04
0.03
0.02
0.01

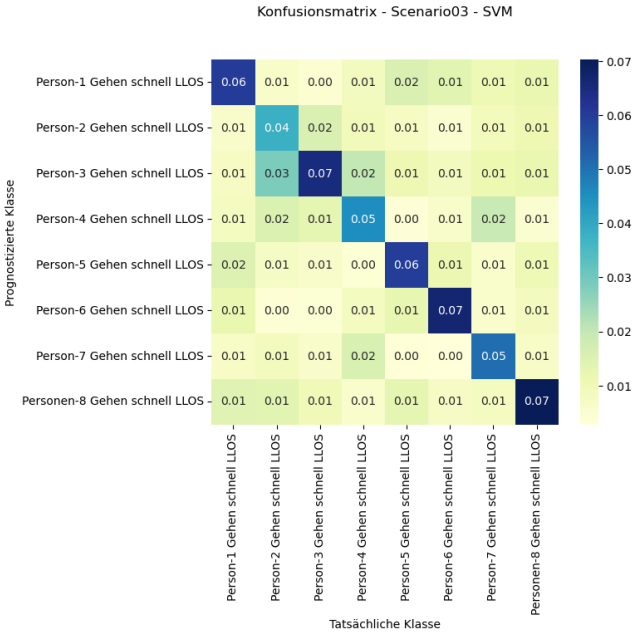

Konfusionsmatrix - Scenario02 - SVM
Prognostizierte Klasse
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
0.06 0.01 0.01 0.01 0.01 0.01 0.01 0.01
0.01 0.06 0.01 0.03 0.01 0.01 0.00 0.01
0.01 0.01 0.05 0.01 0.01 0.01 0.01 0.01
0.01 0.03 0.02 0.04 0.01 0.00 0.01 0.01
0.01 0.01 0.01 0.01 0.06 0.01 0.01 0.01
0.01 0.00 0.01 0.00 0.01 0.07 0.01 0.01
0.01 0.00 0.01 0.01 0.01 0.00 0.07 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.07
Person-1 Gehen normal LLOS
Person-2 Gehen normal LLOS
Person-3 Gehen normal LLOS
Person-4 Gehen normal LLOS
Person-5 Gehen normal LLOS
Person-6 Gehen normal LLOS
Person-7 Gehen normal LLOS
Personen-8 Gehen normal LLOS
Tatsächliche Klasse
0.07
0.06
0.05
0.04
0.03
0.02
0.01

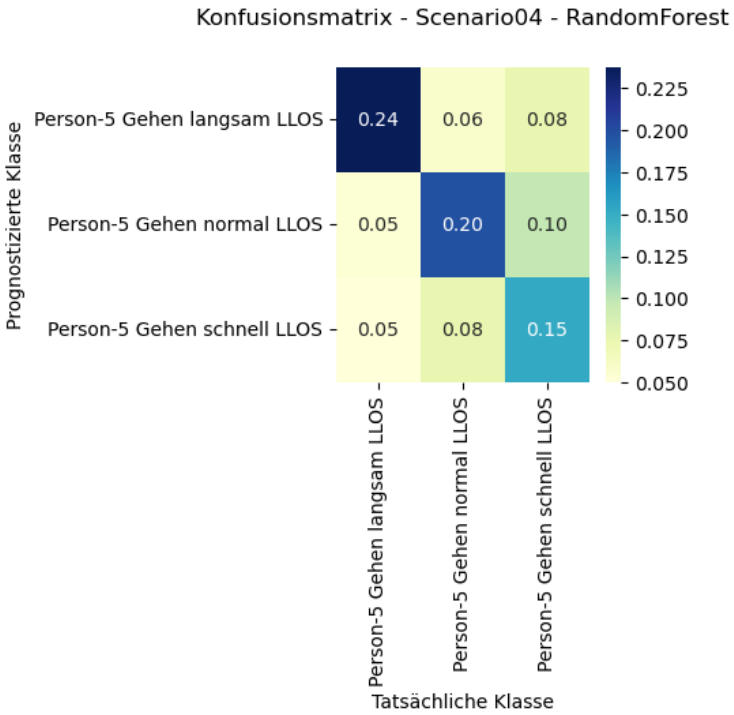

Konfusionsmatrix - Scenario03 - RandomForest
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen schnell LLOS
Person-2 Gehen schnell LLOS
Person-3 Gehen schnell LLOS
Person-4 Gehen schnell LLOS
Person-5 Gehen schnell LLOS
Person-6 Gehen schnell LLOS
Person-7 Gehen schnell LLOS
Personen-8 Gehen schnell LLOS
0.06 0.01 0.00 0.01 0.02 0.01 0.01 0.01
0.01 0.04 0.02 0.01 0.01 0.00 0.01 0.01
0.01 0.03 0.06 0.02 0.01 0.01 0.01 0.01
0.01 0.02 0.01 0.05 0.00 0.01 0.02 0.01
0.02 0.01 0.01 0.00 0.06 0.01 0.01 0.01
0.01 0.01 0.01 0.01 0.01 0.07 0.01 0.01
0.01 0.01 0.01 0.02 0.00 0.00 0.06 0.01
0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.07
0.07
0.06
0.05
0.04
0.03
0.02
0.01

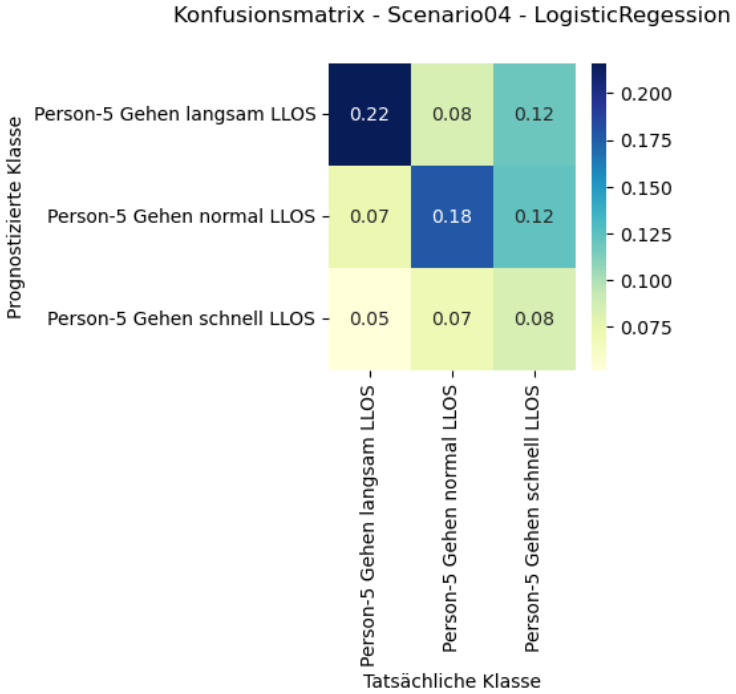

Konfusionsmatrix - Scenario03 - LogisticRegession
Prognostizierte Klasse
Tatsächliche Klasse
Person-1 Gehen schnell LLOS
Person-2 Gehen schnell LLOS
Person-3 Gehen schnell LLOS
Person-4 Gehen schnell LLOS
Person-5 Gehen schnell LLOS
Person-6 Gehen schnell LLOS
Person-7 Gehen schnell LLOS
Personen-8 Gehen schnell LLOS
0.04 0.01 0.01 0.01 0.02 0.01 0.02 0.02
0.01 0.02 0.01 0.01 0.01 0.01 0.01 0.01
0.01 0.01 0.02 0.01 0.01 0.01 0.02 0.01
0.01 0.01 0.01 0.02 0.01 0.01 0.01 0.01
0.01 0.02 0.02 0.01 0.02 0.01 0.02 0.01
0.01 0.02 0.02 0.02 0.02 0.05 0.02 0.01
0.01 0.01 0.01 0.01 0.01 0.00 0.01 0.00
0.03 0.03 0.03 0.02 0.03 0.02 0.02 0.06
0.06
0.05
0.04
0.03
0.02
0.01

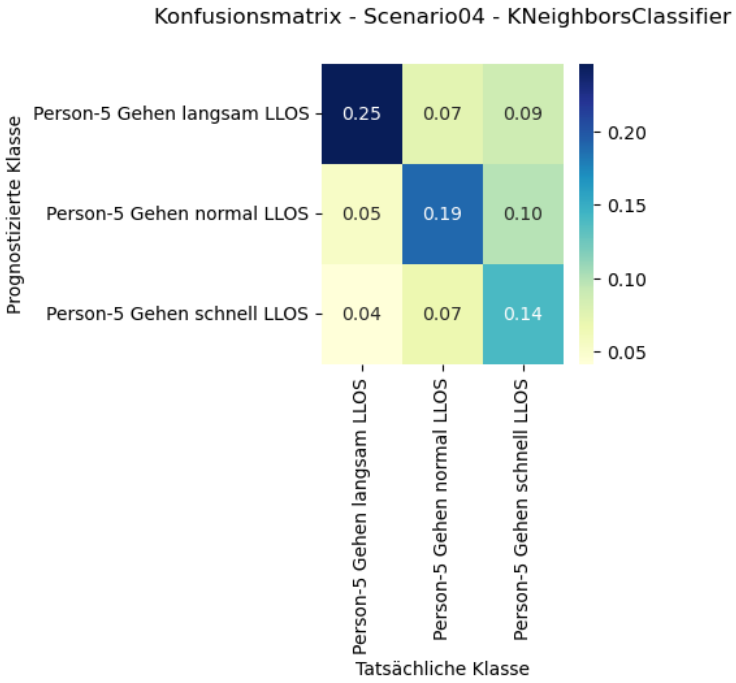

Konfusionsmatrix - Scenario03 - KNeighborsClassifier
Person-1 Gehen schnell LLOS 0.07 0.01 0.01 0.01 0.02 0.02 0.02 0.02
Person-2 Gehen schnell LLOS 0.01 0.05 0.03 0.02 0.01 0.01 0.02 0.02
Person-3 Gehen schnell LLOS 0.01 0.02 0.06 0.02 0.01 0.01 0.01 0.01
Person-4 Gehen schnell LLOS 0.01 0.01 0.01 0.04 0.01 0.01 0.02 0.01
Person-5 Gehen schnell LLOS 0.01 0.01 0.01 0.01 0.05 0.01 0.01 0.01
Person-6 Gehen schnell LLOS 0.01 0.00 0.00 0.01 0.01 0.06 0.01 0.01
Person-7 Gehen schnell LLOS 0.01 0.01 0.01 0.01 0.00 0.00 0.04 0.01
Personen-8 Gehen schnell LLOS 0.01 0.01 0.01 0.00 0.01 0.00 0.01 0.05
Prognostizierte Klasse
Person-1 Gehen schnell LLOS
Person-2 Gehen schnell LLOS
Person-3 Gehen schnell LLOS
Person-4 Gehen schnell LLOS
Person-5 Gehen schnell LLOS
Person-6 Gehen schnell LLOS
Person-7 Gehen schnell LLOS
Personen-8 Gehen schnell LLOS
Tatsächliche Klasse
0.06
0.05
0.04
0.03
0.02
0.01

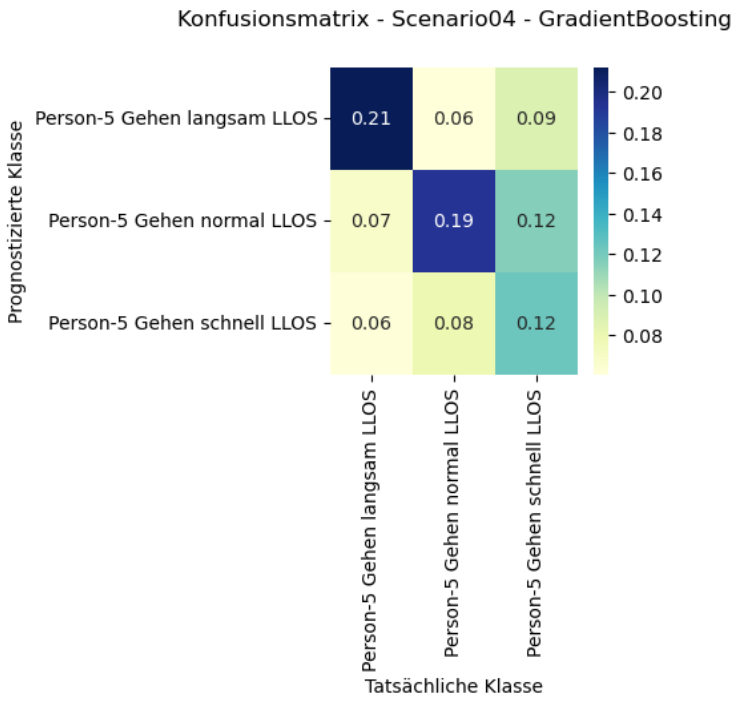

Konfusionsmatrix - Scenario03 - GradientBoosting
Person-1 Gehen schnell LLOS 0.05 0.01 0.01 0.01 0.02 0.01 0.01 0.02
Person-2 Gehen schnell LLOS 0.01 0.02 0.01 0.01 0.01 0.01 0.01 0.01
Person-3 Gehen schnell LLOS 0.01 0.03 0.05 0.02 0.02 0.02 0.01 0.02
Person-4 Gehen schnell LLOS 0.01 0.02 0.02 0.04 0.00 0.01 0.02 0.01
Person-5 Gehen schnell LLOS 0.02 0.01 0.01 0.01 0.05 0.01 0.01 0.02
Person-6 Gehen schnell LLOS 0.01 0.01 0.01 0.01 0.01 0.05 0.01 0.01
Person-7 Gehen schnell LLOS 0.01 0.01 0.01 0.02 0.00 0.00 0.04 0.01
Personen-8 Gehen schnell LLOS 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.06
Prognostizierte Klasse
Person-1 Gehen schnell LLOS
Person-2 Gehen schnell LLOS
Person-3 Gehen schnell LLOS
Person-4 Gehen schnell LLOS
Person-5 Gehen schnell LLOS
Person-6 Gehen schnell LLOS
Person-7 Gehen schnell LLOS
Personen-8 Gehen schnell LLOS
Tatsächliche Klasse
0.05
0.04
0.03
0.02
0.01

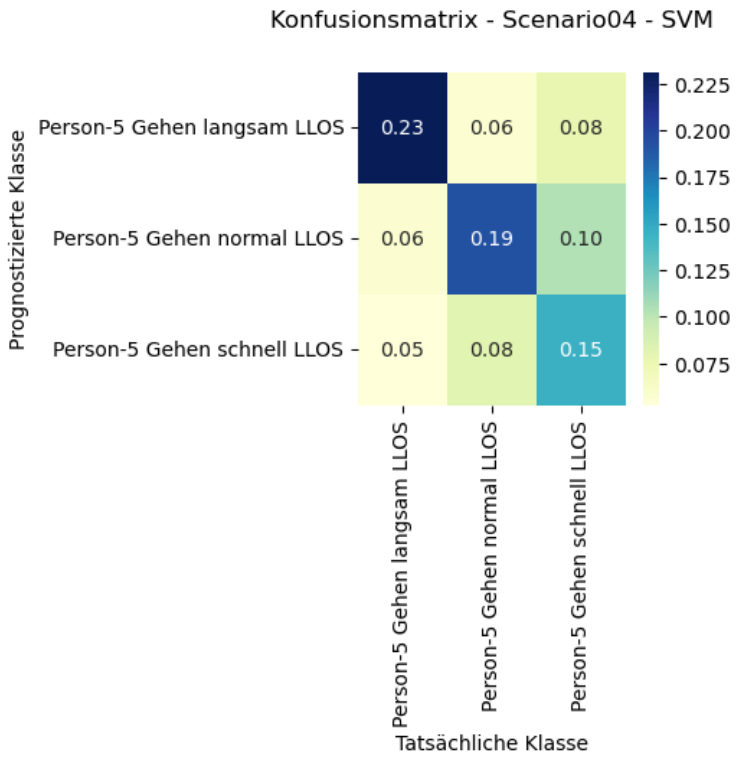

Konfusionsmatrix - Scenario03 - SVM
Person-1 Gehen schnell LLOS
Person-2 Gehen schnell LLOS
Person-3 Gehen schnell LLOS
Person-4 Gehen schnell LLOS
Person-5 Gehen schnell LLOS
Person-6 Gehen schnell LLOS
Person-7 Gehen schnell LLOS
Personen-8 Gehen schnell LLOS
Prognostizierte Klasse
Tatsächliche Klasse
0.07
0.06
0.05
0.04
0.03
0.02
0.01

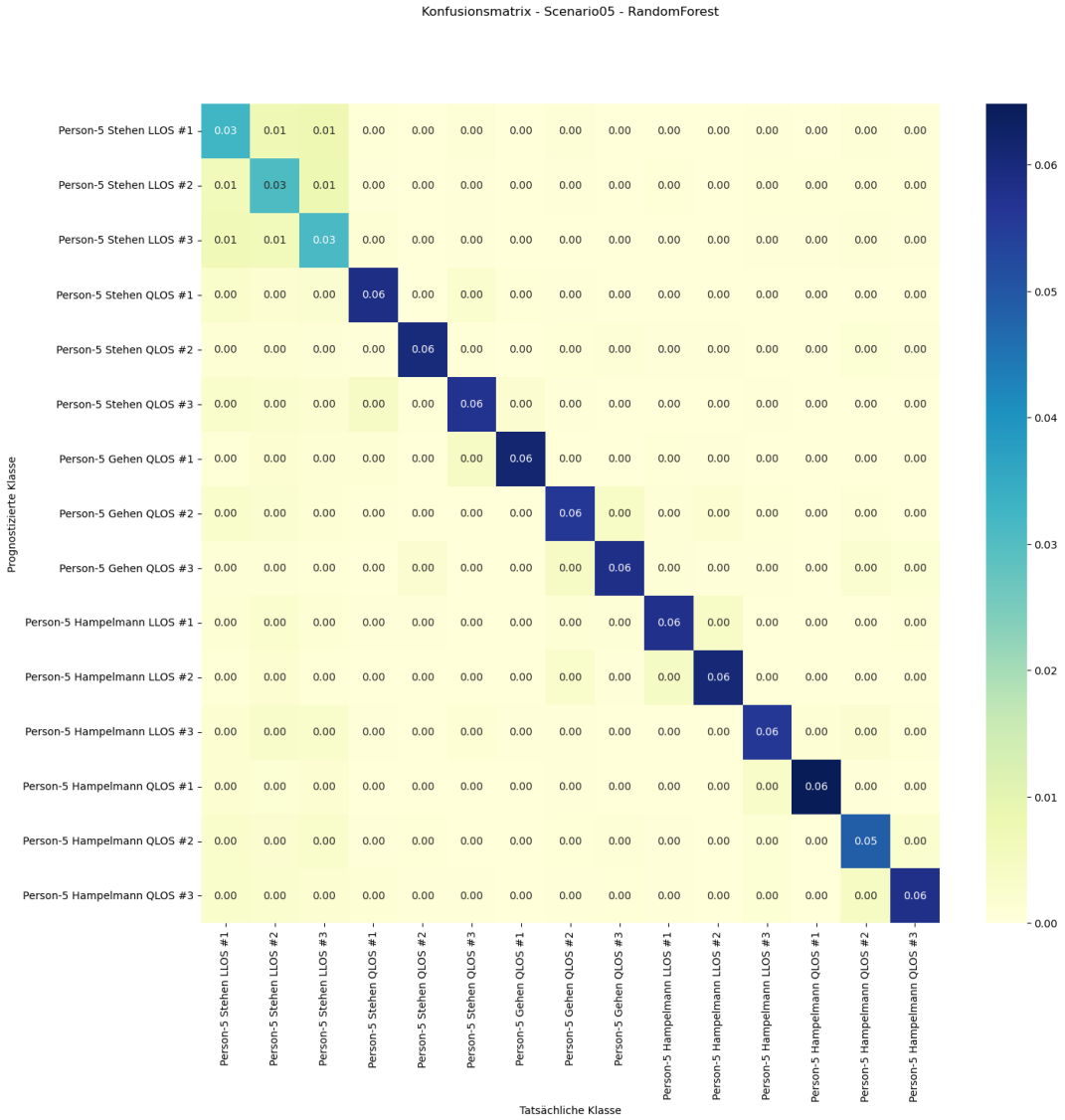

Konfusionsmatrix - Scenario04 - RandomForest
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Prognostizierte Klasse
Tatsächliche Klasse
0.24
0.06
0.08
0.05
0.20
0.10
0.05
0.08
0.15
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050

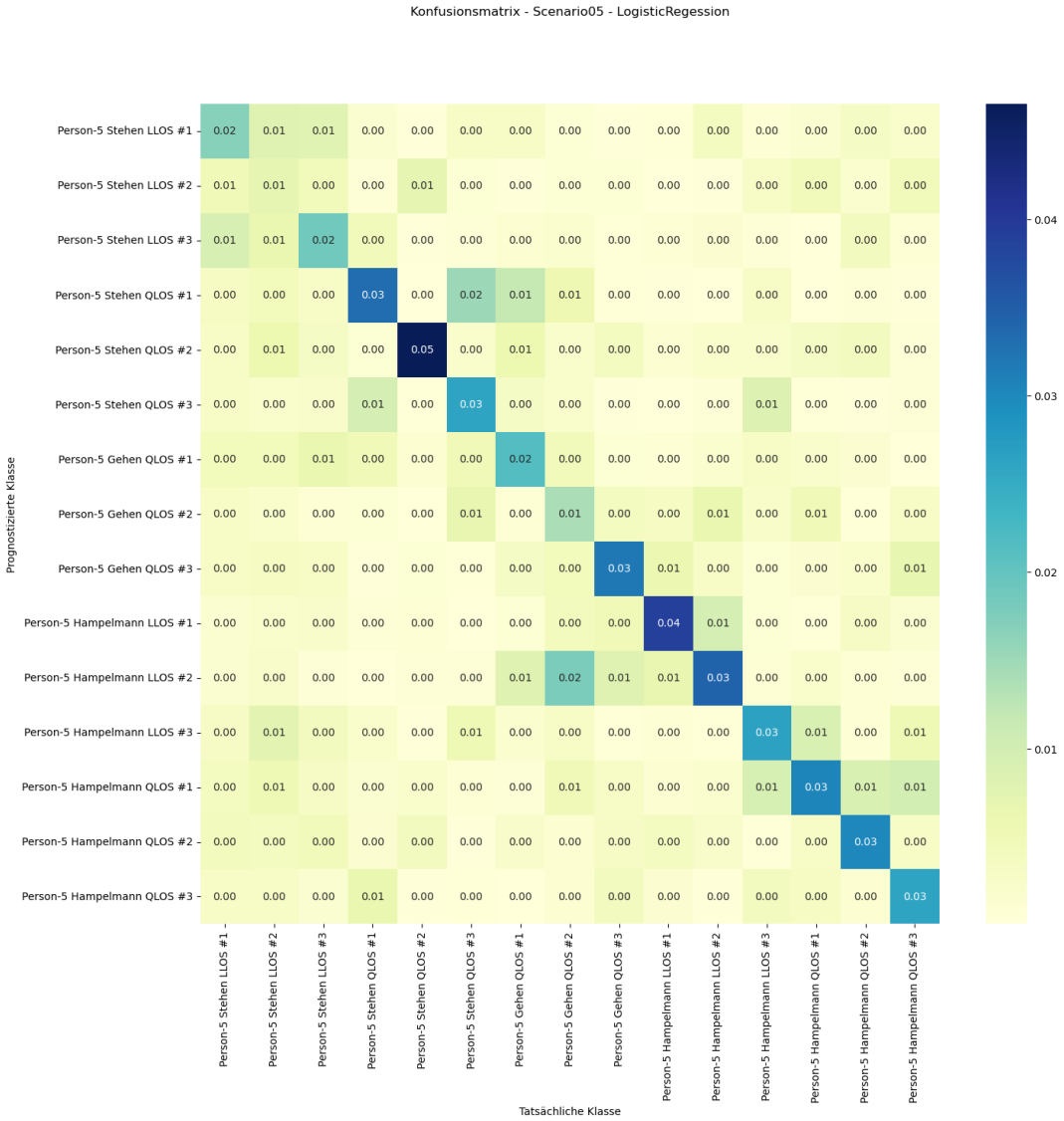

Konfusionsmatrix - Scenario04 - LogisticRegession
Prognostizierte Klasse
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
0.22
0.08
0.12
0.07
0.18
0.12
0.05
0.07
0.08
0.200
0.175
0.150
0.125
0.100
0.075
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Tatsächliche Klasse

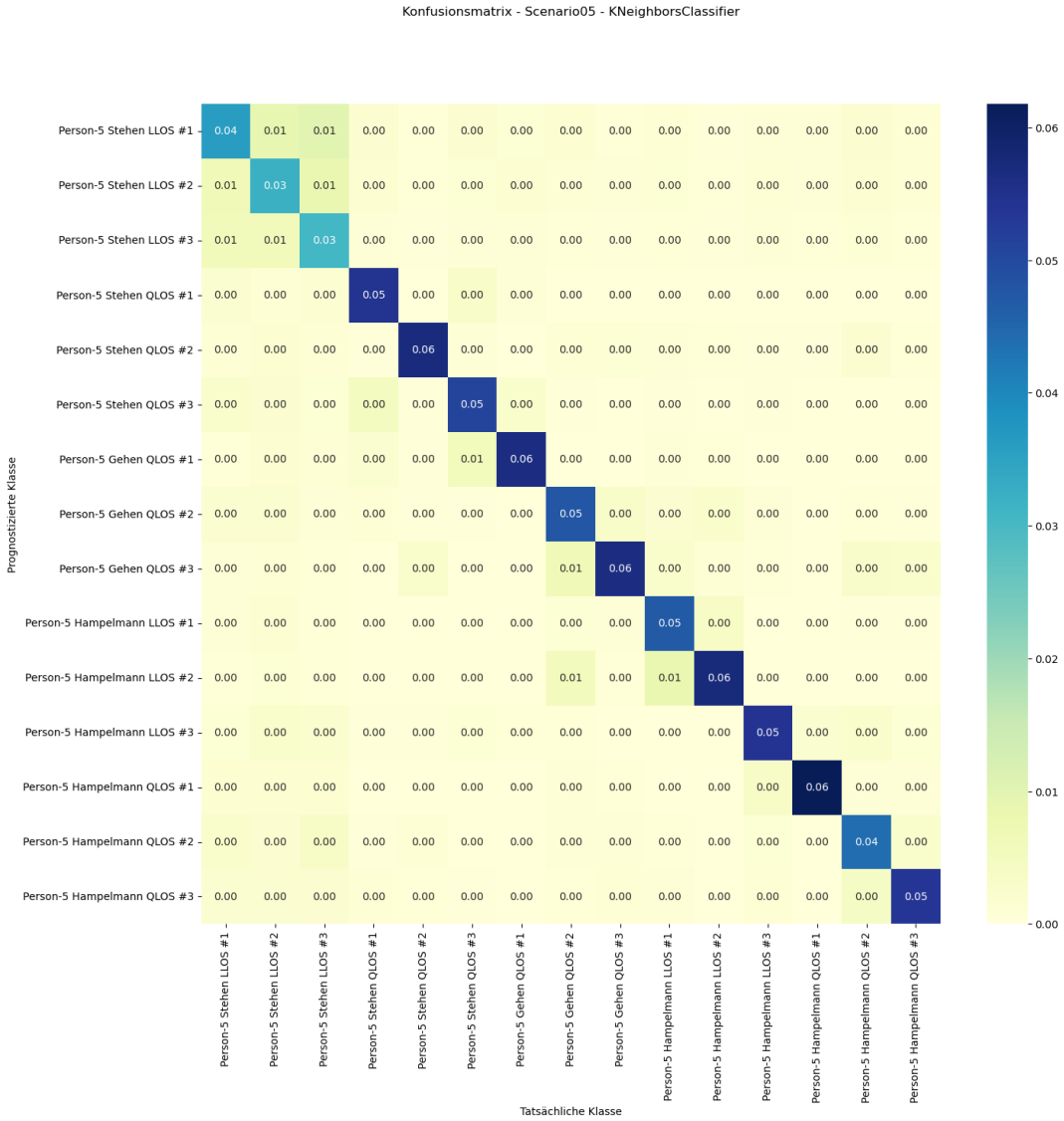

Konfusionsmatrix - Scenario04 - KNeighborsClassifier
Prognostizierte Klasse
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
0.25
0.07
0.09
0.05
0.19
0.10
0.04
0.07
0.14
0.20
0.15
0.10
0.05
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Tatsächliche Klasse

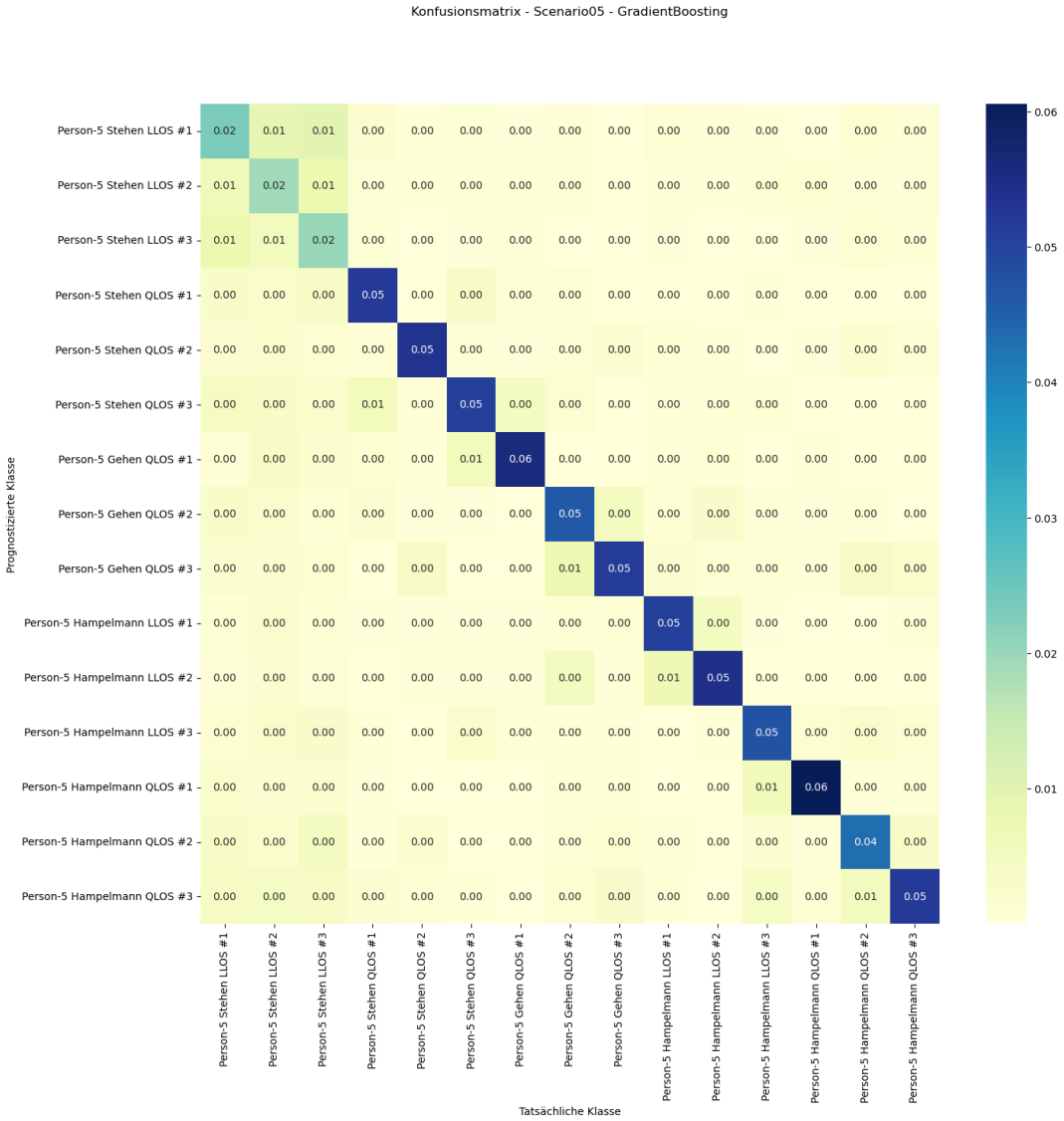

Konfusionsmatrix - Scenario04 - GradientBoosting
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Prognostizierte Klasse
0.21
0.06
0.09
0.07
0.19
0.12
0.06
0.08
0.12
0.20
0.18
0.16
0.14
0.12
0.10
0.08
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Tatsächliche Klasse

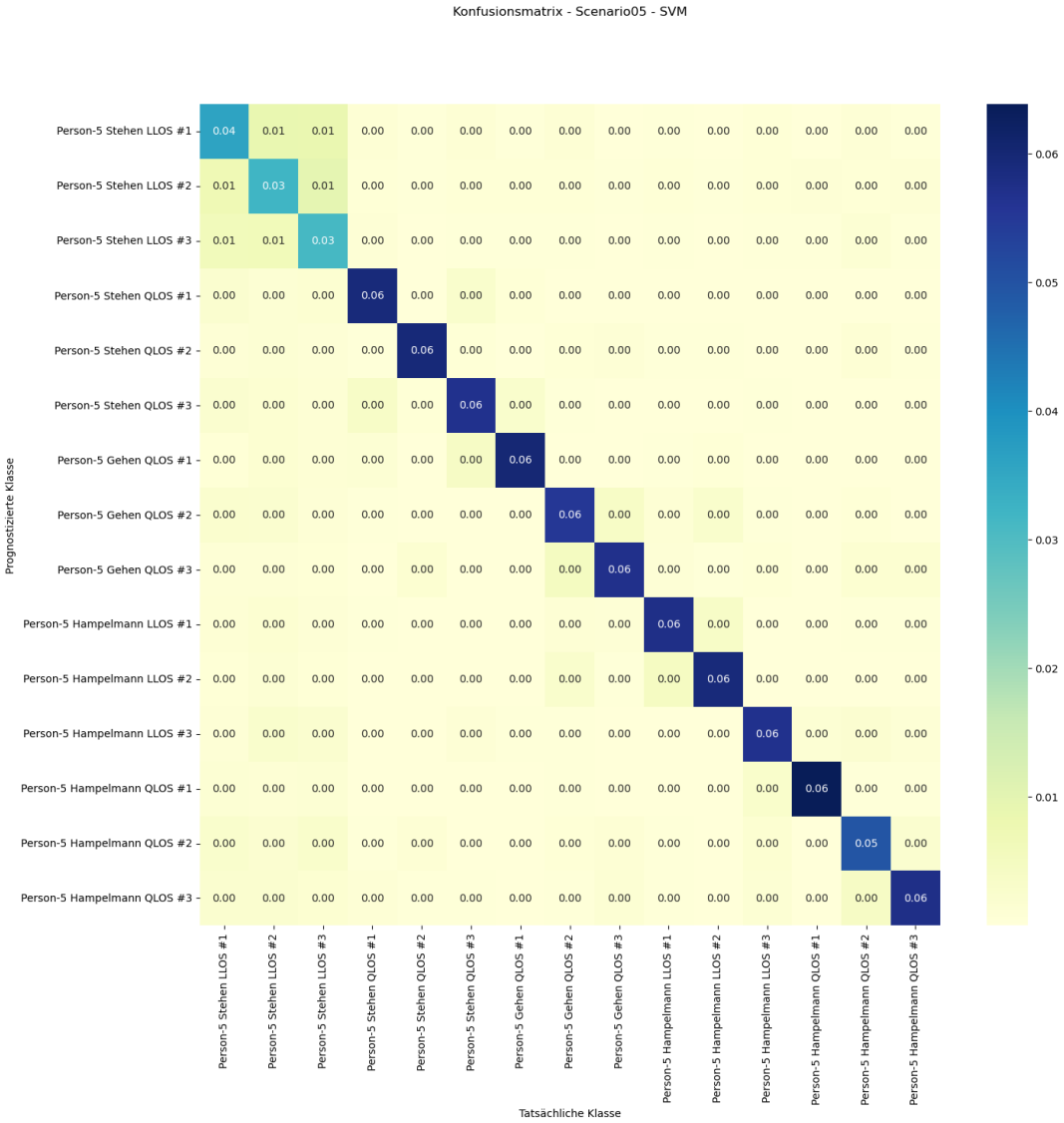

Konfusionsmatrix - Scenario04 - SVM
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Prognostizierte Klasse
0.23
0.06
0.08
0.06
0.19
0.10
0.05
0.08
0.15
0.225
0.200
0.175
0.150
0.125
0.100
0.075
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Tatsächliche Klasse

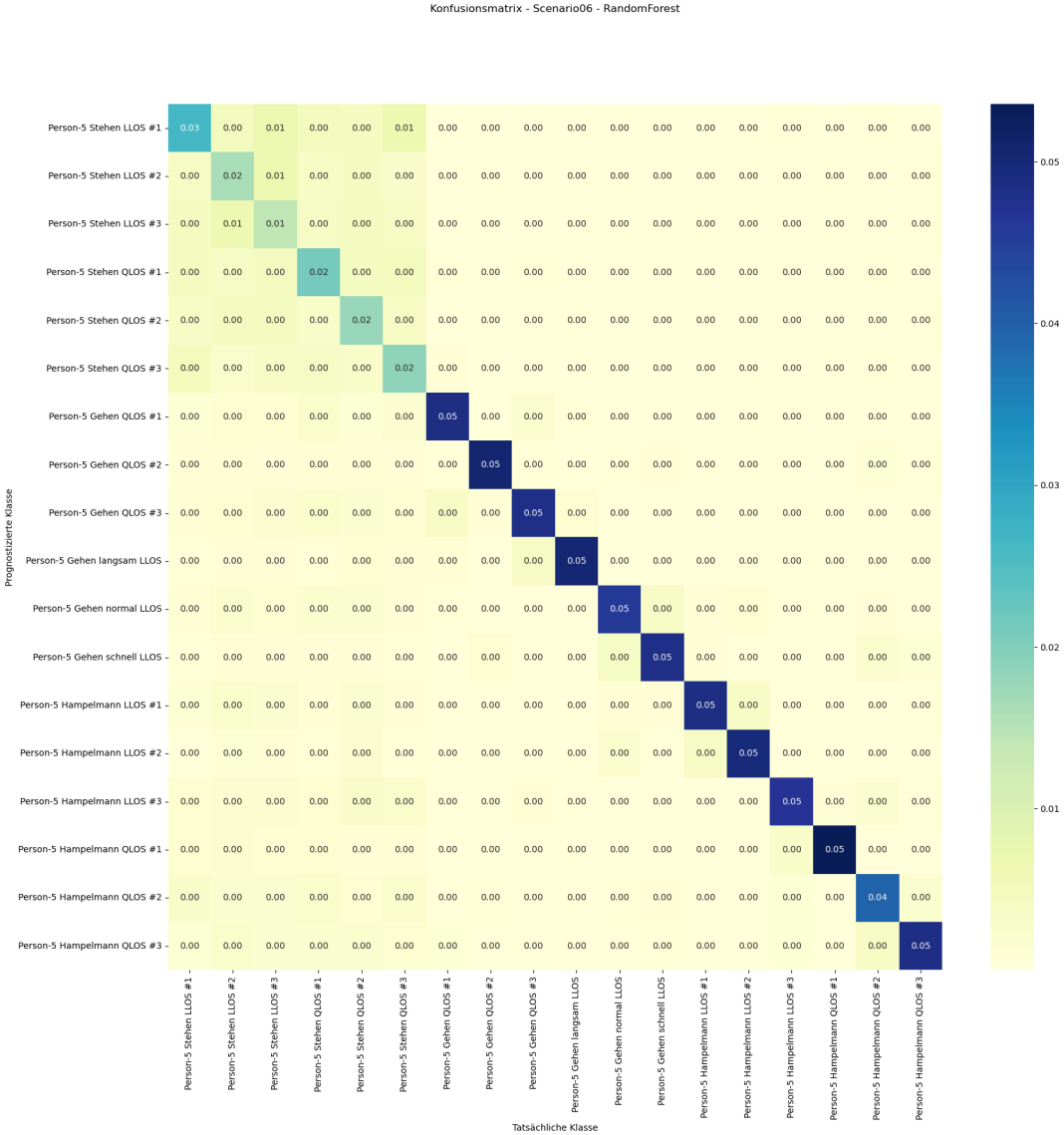

Konfusionsmatrix - Scenario05 - RandomForest
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.06
0.05
0.04
0.03
0.02
0.01
0.00

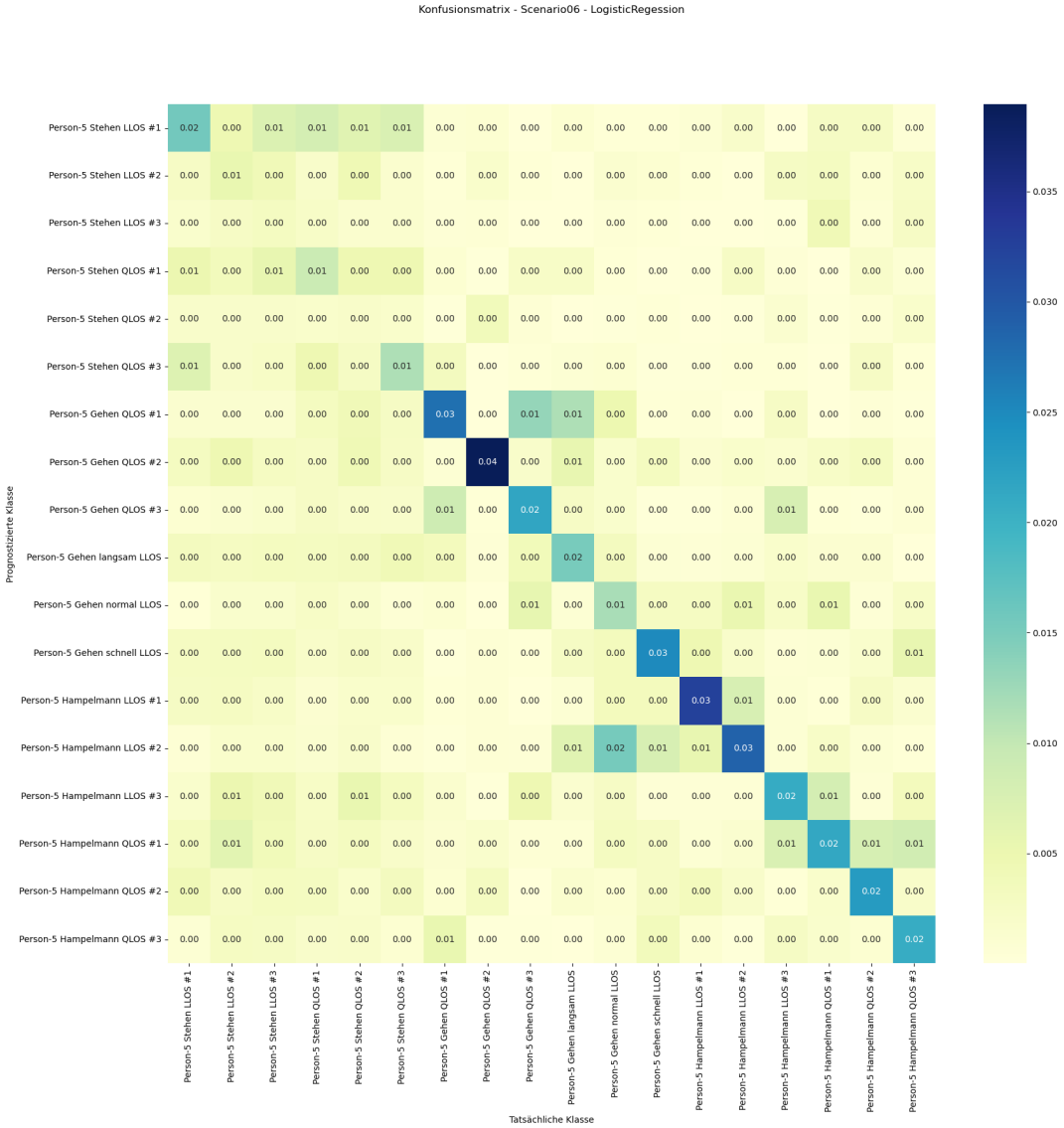

Konfusionsmatrix - Scenario05 - LogisticRegession
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.04
0.03
0.02
0.01

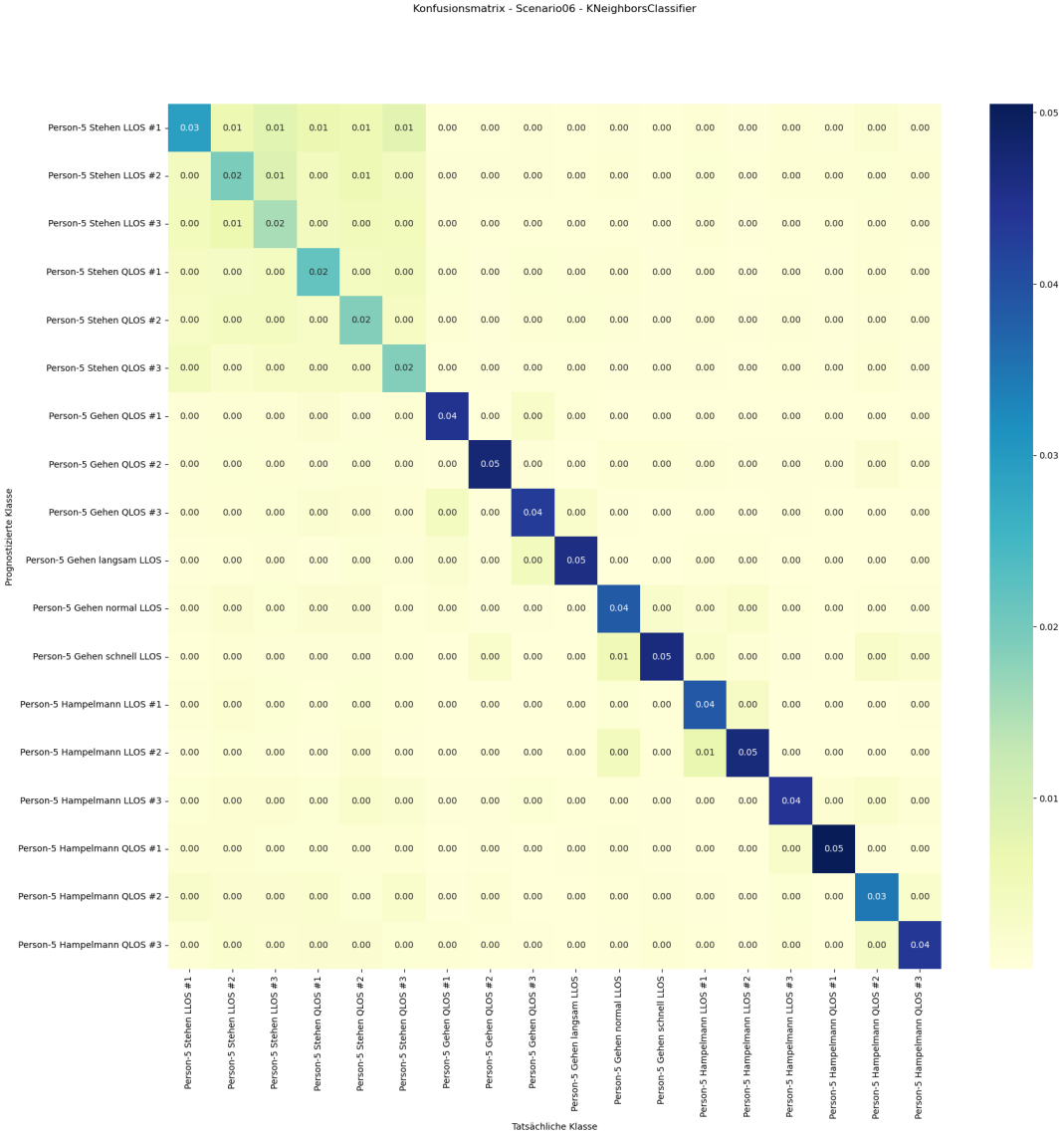

Konfusionsmatrix - Scenario05 - KNeighborsClassifier
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.06
0.05
0.04
0.03
0.02
0.01
0.00

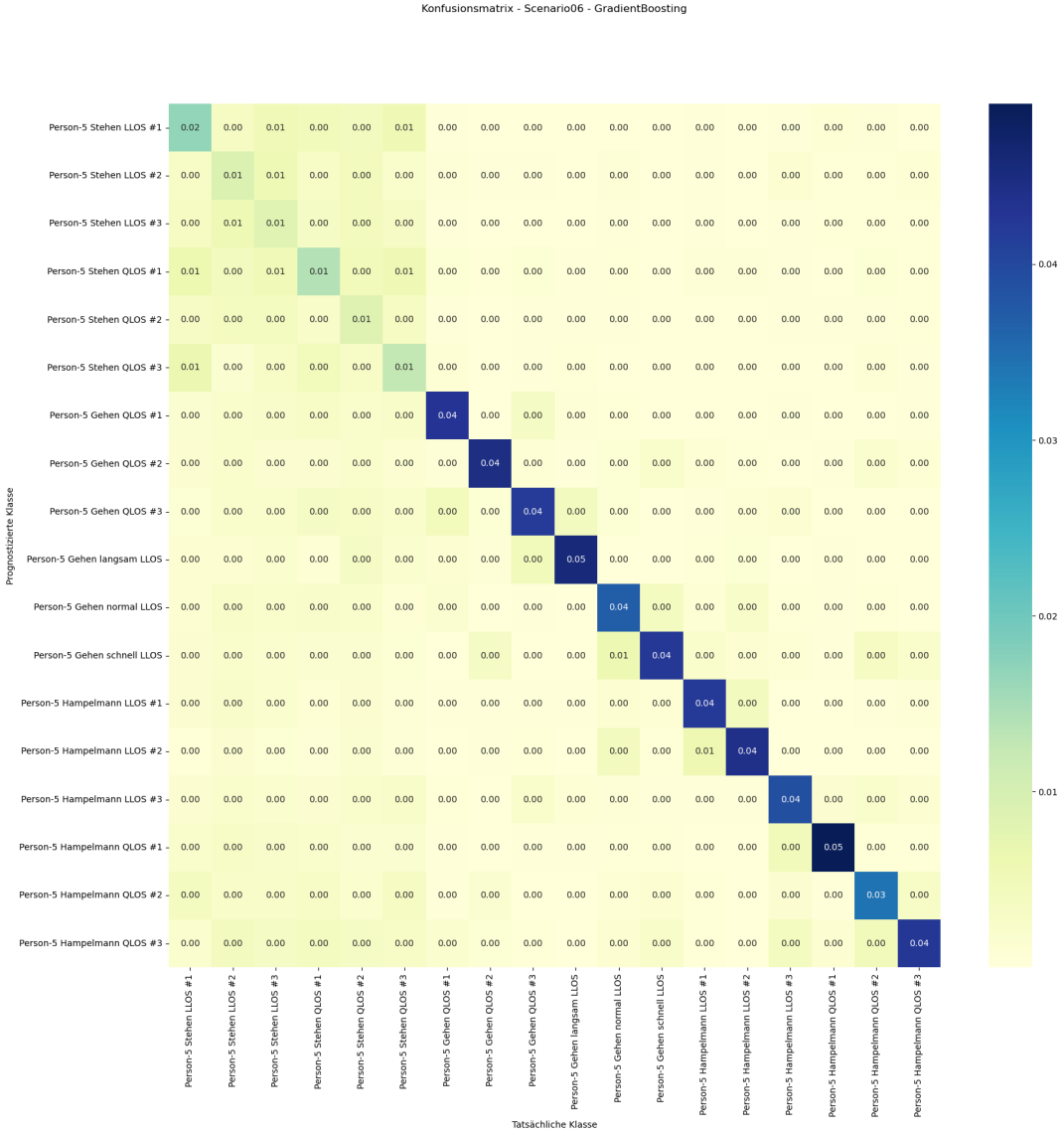

Konfusionsmatrix - Scenario05 - GradientBoosting
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.06
0.05
0.04
0.03
0.02
0.01

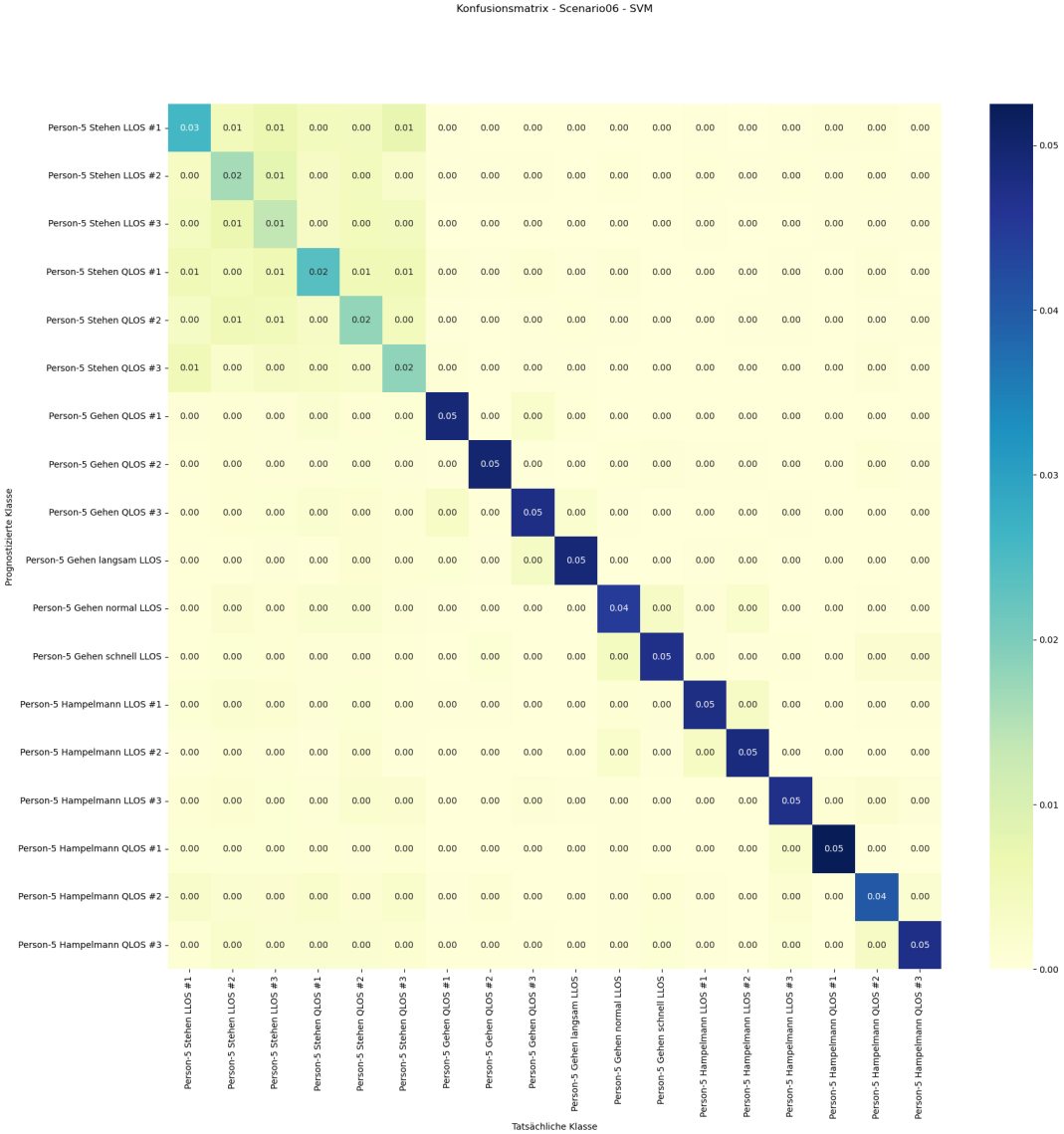

Konfusionsmatrix - Scenario05 - SVM
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.06
0.05
0.04
0.03
0.02
0.01

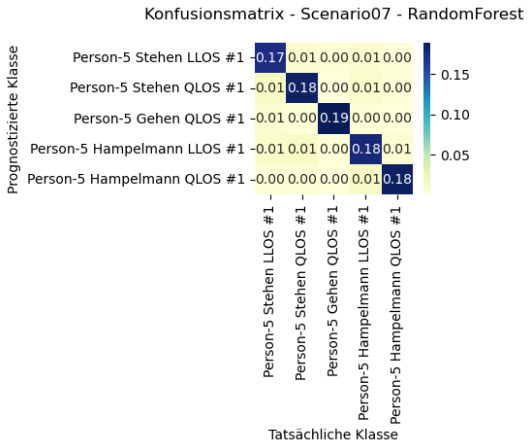

Konfusionsmatrix - Scenario06 - RandomForest
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.05
0.04
0.03
0.02
0.01

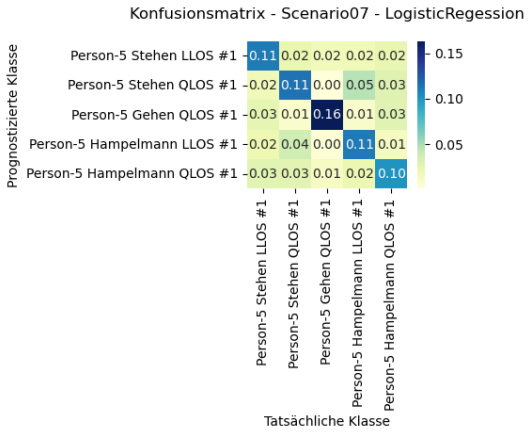

Konfusionsmatrix - Scenario06 - LogisticRegession
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.035
0.030
0.025
0.020
0.015
0.010
0.005

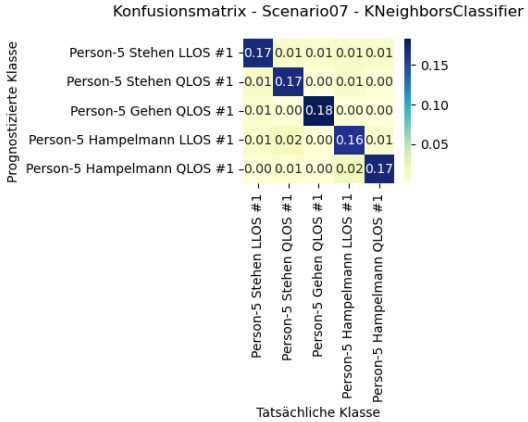

Konfusionsmatrix - Scenario06 - KNeighborsClassifier
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.05
0.04
0.03
0.02
0.01

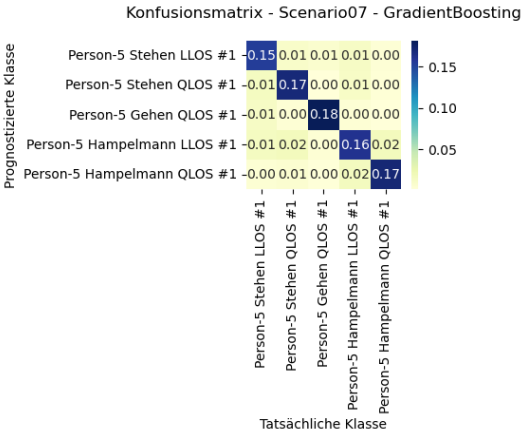

Konfusionsmatrix - Scenario06 - GradientBoosting
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.04
0.03
0.02
0.01

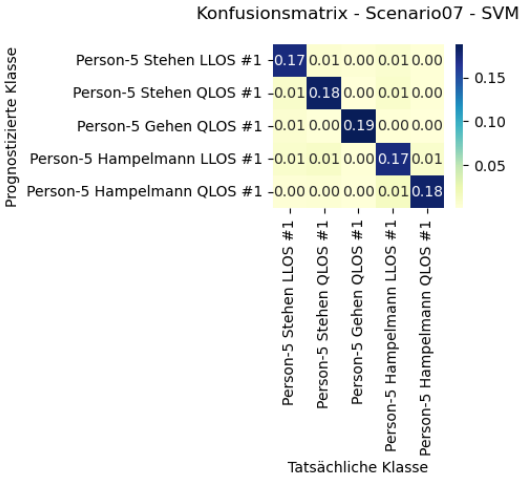

Konfusionsmatrix - Scenario06 - SVM
Prognostizierte Klasse
Tatsächliche Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen LLOS #2
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #1
Person-5 Stehen QLOS #2
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #1
Person-5 Gehen QLOS #2
Person-5 Gehen QLOS #3
Person-5 Gehen langsam LLOS
Person-5 Gehen normal LLOS
Person-5 Gehen schnell LLOS
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #1
Person-5 Hampelmann QLOS #2
Person-5 Hampelmann QLOS #3
0.05
0.04
0.03
0.02
0.01
0.00

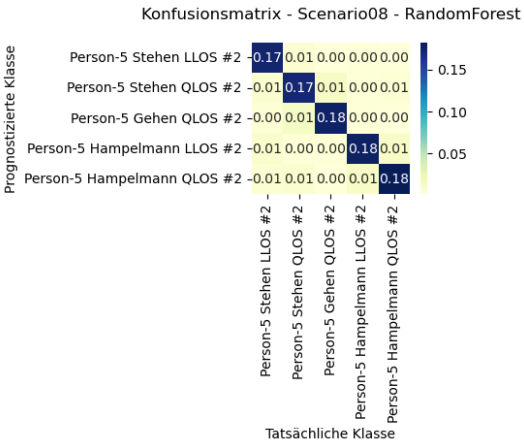

Konfusionsmatrix - Scenario07 - RandomForest
Prognostizierte Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
0.17 0.01 0.00 0.01 0.00
0.01 0.18 0.00 0.01 0.00
0.01 0.00 0.19 0.00 0.00
0.01 0.01 0.00 0.18 0.01
0.00 0.00 0.00 0.01 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
Tatsächliche Klasse

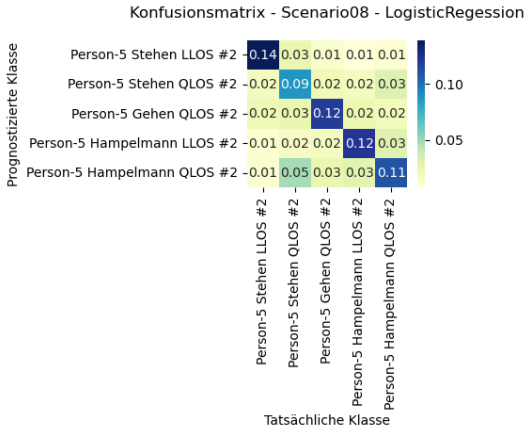

Konfusionsmatrix - Scenario07 - LogisticRegession
Prognostizierte Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
0.11 0.02 0.02 0.02 0.02
0.02 0.11 0.00 0.05 0.03
0.03 0.01 0.16 0.01 0.03
0.02 0.04 0.00 0.11 0.01
0.03 0.03 0.01 0.02 0.10
0.15
0.10
0.05
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
Tatsächliche Klasse

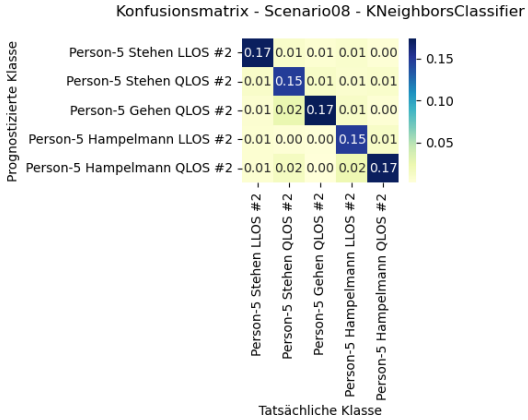

Konfusionsmatrix - Scenario07 - KNeighborsClassifier
Prognostizierte Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
0.17 0.01 0.01 0.01 0.01
0.01 0.17 0.00 0.01 0.00
0.01 0.00 0.18 0.00 0.00
0.01 0.02 0.00 0.16 0.01
0.00 0.01 0.00 0.02 0.17
0.15
0.10
0.05
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
Tatsächliche Klasse

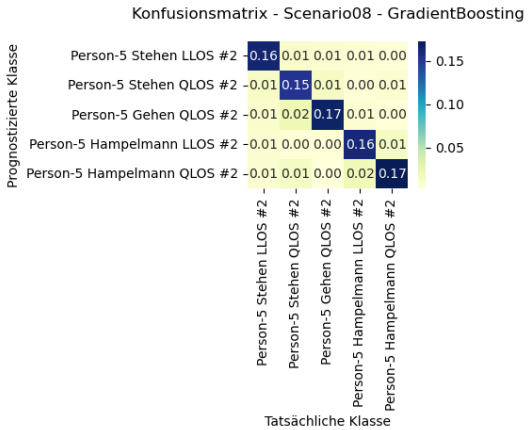

Konfusionsmatrix - Scenario07 - GradientBoosting
Prognostizierte Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
0.15 0.01 0.01 0.01 0.00
0.01 0.17 0.00 0.01 0.00
0.01 0.00 0.18 0.00 0.00
0.01 0.02 0.00 0.16 0.02
0.00 0.01 0.00 0.02 0.17
0.15
0.10
0.05
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
Tatsächliche Klasse

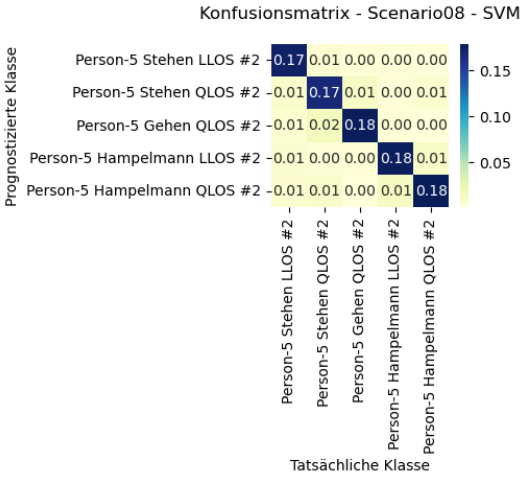

Konfusionsmatrix - Scenario07 - SVM
Prognostizierte Klasse
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
0.17 0.01 0.00 0.01 0.00
0.01 0.18 0.00 0.01 0.00
0.01 0.00 0.19 0.00 0.00
0.01 0.01 0.00 0.17 0.01
0.00 0.00 0.00 0.01 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #1
Person-5 Stehen QLOS #1
Person-5 Gehen QLOS #1
Person-5 Hampelmann LLOS #1
Person-5 Hampelmann QLOS #1
Tatsächliche Klasse

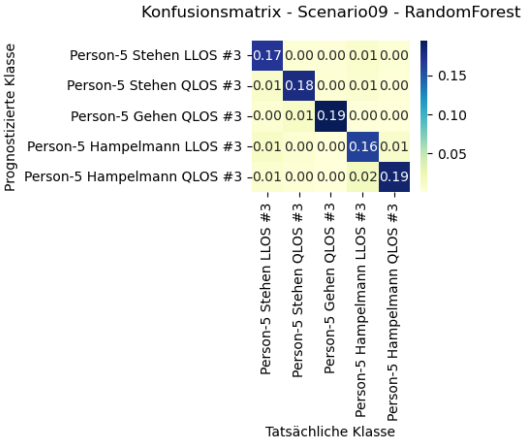

Konfusionsmatrix - Scenario08 - RandomForest
Prognostizierte Klasse
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
0.17 0.01 0.00 0.00 0.00
0.01 0.17 0.01 0.00 0.01
0.00 0.01 0.18 0.00 0.00
0.01 0.00 0.00 0.18 0.01
0.01 0.01 0.00 0.01 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
Tatsächliche Klasse

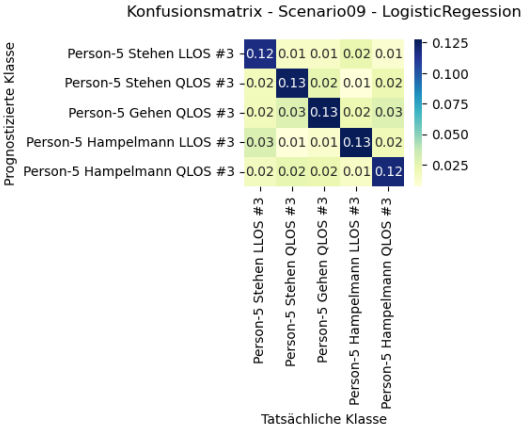

Konfusionsmatrix - Scenario08 - LogisticRegession
Prognostizierte Klasse
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
0.14 0.03 0.01 0.01 0.01
0.02 0.09 0.02 0.02 0.03
0.02 0.03 0.12 0.02 0.02
0.01 0.02 0.02 0.12 0.03
0.01 0.05 0.03 0.03 0.11
0.10
0.05
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
Tatsächliche Klasse

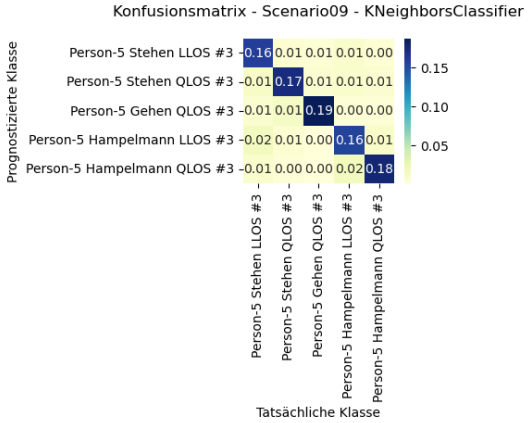

Konfusionsmatrix - Scenario08 - KNeighborsClassifier
Prognostizierte Klasse
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
0.17 0.01 0.01 0.01 0.00
0.01 0.15 0.01 0.01 0.01
0.01 0.02 0.17 0.01 0.00
0.01 0.00 0.00 0.15 0.01
0.01 0.02 0.00 0.02 0.17
0.15
0.10
0.05
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
Tatsächliche Klasse

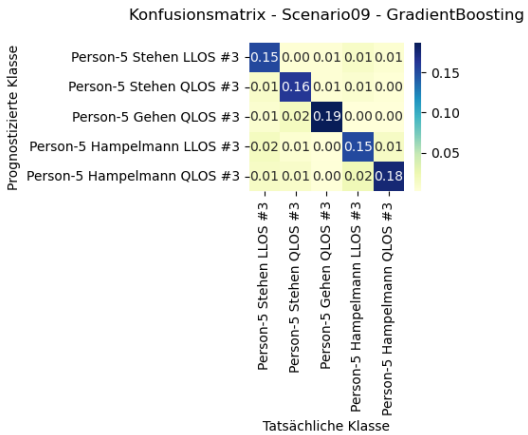

Konfusionsmatrix - Scenario08 - GradientBoosting
Prognostizierte Klasse
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
0.16 0.01 0.01 0.01 0.00
0.01 0.15 0.01 0.00 0.01
0.01 0.02 0.17 0.01 0.00
0.01 0.00 0.00 0.16 0.01
0.01 0.01 0.00 0.02 0.17
0.15
0.10
0.05
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
Tatsächliche Klasse

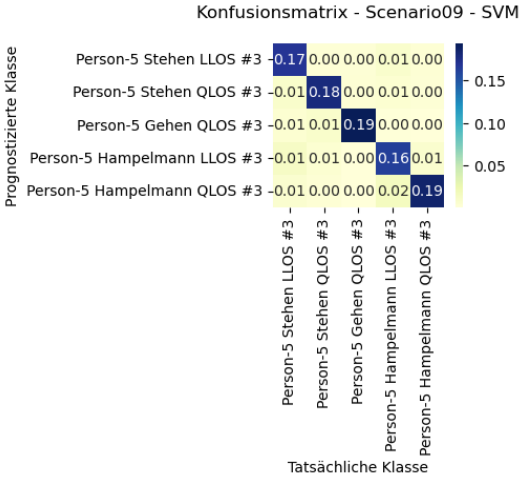

Konfusionsmatrix - Scenario08 - SVM
Prognostizierte Klasse
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
0.17 0.01 0.00 0.00 0.00
0.01 0.17 0.01 0.00 0.01
0.01 0.02 0.18 0.00 0.00
0.01 0.00 0.00 0.18 0.01
0.01 0.01 0.00 0.01 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #2
Person-5 Stehen QLOS #2
Person-5 Gehen QLOS #2
Person-5 Hampelmann LLOS #2
Person-5 Hampelmann QLOS #2
Tatsächliche Klasse

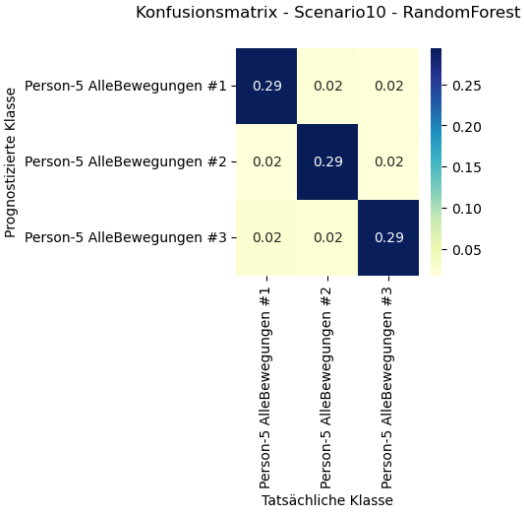

Konfusionsmatrix - Scenario09 - RandomForest
Prognostizierte Klasse
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
0.17 0.00 0.00 0.01 0.00
0.01 0.18 0.00 0.01 0.00
0.00 0.01 0.19 0.00 0.00
0.01 0.00 0.00 0.16 0.01
0.01 0.00 0.00 0.02 0.19
0.15
0.10
0.05
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
Tatsächliche Klasse

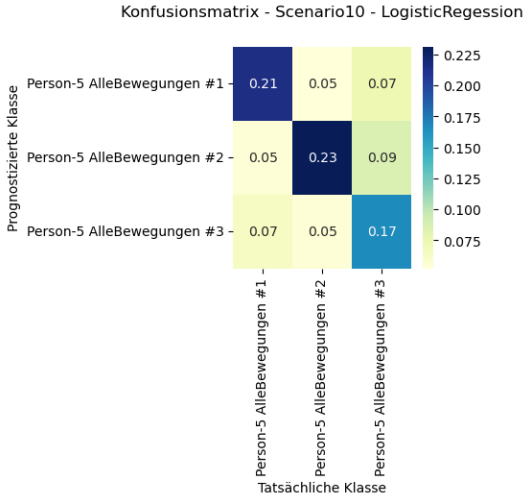

Konfusionsmatrix - Scenario09 - LogisticRegession
Prognostizierte Klasse
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
0.12 0.01 0.01 0.02 0.01
0.02 0.13 0.02 0.01 0.02
0.02 0.03 0.13 0.02 0.03
0.03 0.01 0.01 0.13 0.02
0.02 0.02 0.02 0.01 0.12
0.125
0.100
0.075
0.050
0.025
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
Tatsächliche Klasse

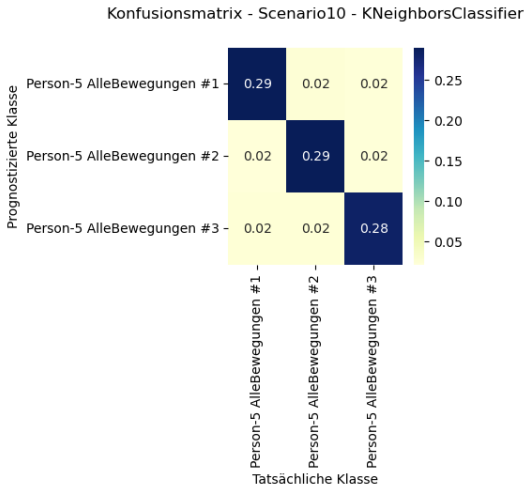

Konfusionsmatrix - Scenario09 - KNeighborsClassifier
Prognostizierte Klasse
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
0.16 0.01 0.01 0.01 0.00
0.01 0.17 0.01 0.01 0.01
0.01 0.01 0.19 0.00 0.00
0.02 0.01 0.00 0.16 0.01
0.01 0.00 0.00 0.02 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
Tatsächliche Klasse

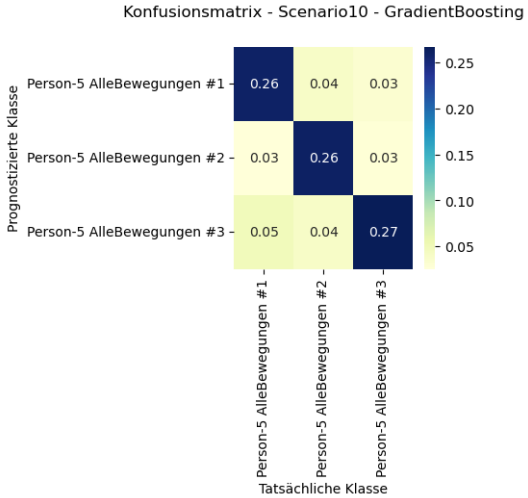

Konfusionsmatrix - Scenario09 - GradientBoosting
Prognostizierte Klasse
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
0.15 0.00 0.01 0.01 0.01
0.01 0.16 0.01 0.01 0.00
0.01 0.02 0.19 0.00 0.00
0.02 0.01 0.00 0.15 0.01
0.01 0.01 0.00 0.02 0.18
0.15
0.10
0.05
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
Tatsächliche Klasse

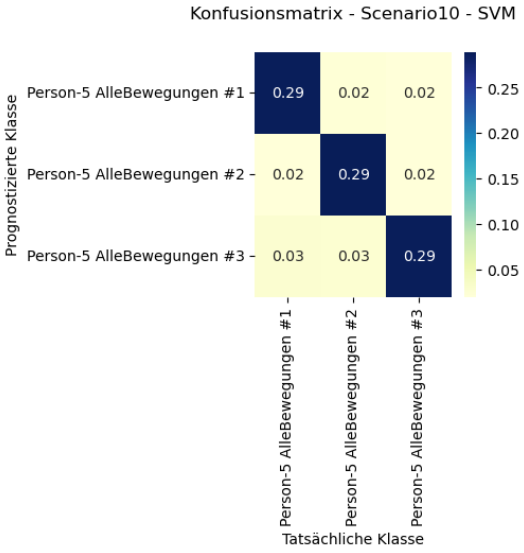

Konfusionsmatrix - Scenario09 - SVM
Prognostizierte Klasse
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
0.17 0.00 0.00 0.01 0.00
0.01 0.18 0.00 0.01 0.00
0.01 0.01 0.19 0.00 0.00
0.01 0.01 0.00 0.16 0.01
0.01 0.00 0.00 0.02 0.19
0.15
0.10
0.05
Person-5 Stehen LLOS #3
Person-5 Stehen QLOS #3
Person-5 Gehen QLOS #3
Person-5 Hampelmann LLOS #3
Person-5 Hampelmann QLOS #3
Tatsächliche Klasse


Konfusionsmatrix - Scenario10 - RandomForest
Prognostizierte Klasse
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
0.29 0.02 0.02
0.02 0.29 0.02
0.02 0.02 0.29
0.25
0.20
0.15
0.10
0.05
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
Tatsächliche Klasse


Konfusionsmatrix - Scenario10 - LogisticRegession
Prognostizierte Klasse
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
0.21
0.05
0.07
0.05
0.23
0.09
0.07
0.05
0.17
0.225
0.200
0.175
0.150
0.125
0.100
0.075
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
Tatsächliche Klasse


Konfusionsmatrix - Scenario10 - KNeighborsClassifier
Prognostizierte Klasse
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
0.29
0.02
0.02
0.02
0.29
0.02
0.02
0.02
0.28
0.25
0.20
0.15
0.10
0.05
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
Tatsächliche Klasse


Konfusionsmatrix - Scenario10 - GradientBoosting
Prognostizierte Klasse
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
0.26
0.04
0.03
0.03
0.26
0.03
0.05
0.04
0.27
0.25
0.20
0.15
0.10
0.05
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
Tatsächliche Klasse


Konfusionsmatrix - Scenario10 - SVM
Prognostizierte Klasse
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
0.29
0.02
0.02
0.02
0.29
0.02
0.03
0.03
0.29
0.25
0.20
0.15
0.10
0.05
Person-5 AlleBewegungen #1
Person-5 AlleBewegungen #2
Person-5 AlleBewegungen #3
Tatsächliche Klasse