HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Frau
Jasmin Kermer

Vorhersage des Erkrankungsrisikos bei Kühen mit maschinellen Lernverfahren anhand von Milchleistungsprüfungsdaten.

2022

Fakultät **Angewandte Computer- und Biowissenschaften**

BACHELORARBEIT

Vorhersage des Erkrankungsrisikos bei Kühen mit maschinellen Lernverfahren anhand von Milchleistungsprüfungsdaten.

Autorin:
Jasmin Kermer

Studiengang:
Bachelor Biotechnologie

Seminargruppe:
BT19wB-B

Erstprüfer:
Prof. Dr. rer. nat. habil. Thomas Villmann

Mittweida, 15. August 2022

Bibliografische Angaben

Kermer, Jasmin: Vorhersage des Erkrankungsrisikos bei Kühen mit maschinellen Lernverfahren anhand von Milchleistungsprüfungsdaten., 68 Seiten, 34 Abbildungen, Hochschule Mittweida, University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften

Englischer Titel: *Prediction of cow diseases risk by the analysis of milk performance test data using machine learning methods.*

Bachelorarbeit, 2022

Satz: LaTeX

In regelmäßigen Abständen werden bei Milchkühen in Deutschland Daten über die Zusammensetzung der Milch erhoben, um eine gleichbleibende Qualität sicherstellen zu können. Gleichzeitig dienen die Milchinhaltsstoffe als erste Indikatoren für eine Veränderung des Stoffwechsels der Kuh und ein damit einhergehend erhöhtes Erkrankungsrisiko. Aus diesem Grund wird in dieser Arbeit untersucht, ob es möglich ist, Vorhersagen über Erkrankungen bei Milchkühen anhand dieser Milchleistungsprüfungsdaten zu treffen. Dafür werden maschinelle Lernverfahren angewendet, im Speziellen Multi-Label- und binäre Klassifikationsverfahren. Die genutzten Klassifikatoren umfassen Multi-Layer Perzeptrone, Naive Bayes-Klassifikatoren sowie Support Vector Machines mit verschiedenen Kernels. Die Vorhersagen werden mit Konfusionsmatrizen und den dazugehörigen Evaluationsmaßen ausgewertet und verglichen.

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

1 Einleitung

In Deutschland werden jährlich 32.5 Millionen Tonnen Milch in Form von Trinkmilch, Joghurt, Käse und anderen Milchprodukten hergestellt (Eurostat, 2020). Um diesem Bedarf gerecht zu werden und gleichzeitig die Gesundheit der Kühe aufrechtzuerhalten, die diese Milch herstellen, werden in regelmäßigen Abständen sogenannte Milchleistungsprüfungen (kurz: MLP) durchgeführt. Diese beinhalten zahlreiche Informationen zur Milchmenge und den wichtigsten Milchinhaltsstoffen, aus denen erste Anhaltspunkte zur gesundheitlichen Verfassung und zum Wohlbefinden der Kühe abgeleitet werden können.
Da die Tiergesundheit eng mit dem betriebswirtschaftlichen Erfolg verknüpft ist, sind diese regelmäßigen Kontrollen unabdinglich. Denn eine kranke Kuh führt durch Tierarztkosten, zusätzlichen Pflegeaufwand und Erlöseinbußen durch geringere Milchleistung schnell zu zusätzlichen Kosten (Galligan, 2006). Gleichzeitig sorgen die geringen Milchpreise und immer größer werdenden Betriebe dafür, dass immer weniger Zeit pro Tier aufgewendet werden kann (Statistisches Bundesamt, 2021).
Können die Daten aus den Milchleistungsprüfungen genutzt werden, um das mögliche Erkrankungsrisiko einer Kuh abzuschätzen? Das soll im Rahmen dieser Bachelorarbeit näher untersucht werden.

1.1 Motivation und Zielsetzung

Etwa ein Drittel der Milchkühe in Deutschland werden jedes Jahr geschlachtet. Mehr als die Hälfte dieser Abgänge ist krankheitlich bedingt. Fruchtbarkeitsstörungen, Eutererkrankungen sowie Erkrankungen der Klauen oder Gliedmaßen bilden dabei den Hauptanteil (Bauer, Martens und Thöne-Reineke, 2021).
Im Bereich Eutergesundheit und Fruchtbarkeit wird die sogenannte integrierte tierärztliche Bestandsbetreuung (ITB) am häufigsten in Anspruch genommen. Dabei werden die verschiedenen Faktoren, die einen Einfluss auf die Herdengesundheit, Herdenfruchtbarkeit oder die Milchleistung haben, von Tierärzten untersucht und angepasst (PraeRi, 2020). Die Kosten für solche und ähnliche Untersuchungen könnten minimiert werden, wenn den Betrieben ein Werkzeug an die Hand gegeben wird, mit dem sie selbst Erkrankungsrisiko eines Tieres abschätzen können.

Ziel dieser Arbeit ist die Entwicklung eines maschinellen Lernworkflows, mit dem anhand der Daten der Milchleistungsprüfungen eine Vorhersage darüber getroffen werden kann, welche Kuh in näherer Zukunft ein erhöhtes Risiko aufweist, an einer bestimmten Krankheit zu erkranken. Damit sollen frühzeitig entgegensteuernde Maßnahmen getroffen werden können, die zu einer Verbesserung der Tiergesundheit führen.

1.2 Problemstellung

Die konkrete Problemstellung der Arbeit besteht darin, anhand der letzten Milchleistungsprüfung einer Laktationsphase vorherzusagen, ob es in der darauf folgenden Laktationsphase in den ersten 30 bzw 100 Tagen zu einer Erkrankung kommt.
Wie in Kapitel 2 näher erläutert wird, beschreibt eine Laktationsphase den Zeitraum, in dem eine Kuh Milch gibt. Sie beginnt mit der Geburt des Kalbs und dauert etwa ein Jahr. Etwa elf Mal während einer Laktation werden Milchleistungsprüfungen in einem Abstand von 30 Tagen durchgeführt. Zwischen zwei Laktationen liegt ein Zeitraum von circa zwei Monaten, die sogenannte Trockenstehphase. Wenige Wochen vor und während der Trockenphase kommt es zu starken Stoffwechselveränderungen, die zu einem erhöhtem Erkrankungsrisiko führen können.
Es wird angenommen, dass sich die ersten Anzeichen dieser Veränderungen bereits innerhalb des Zeitraums der letzten Milchleistungsprüfung einer Laktation zeigen können. Aus diesem Grund soll diese letzte MLP genutzt werden, um Vorhersagen über auftretende Erkrankungen zu Beginn der nachfolgenden Laktation zu treffen.
Eine Visualisierung der Problemstellung ist in Abbildung A.1 dargestellt.

1.3 Aufbau der Arbeit

Im ersten Teil der Arbeit werden die biologischen Zusammenhänge zwischen den Veränderungen der Milchinhaltsstoffe und dem Erkrankungsrisiko erläutert. Daran schließt sich ein Überblick über bisherige Untersuchungen an, die sich ebenfalls mit der Vorhersage von Erkrankungen auf Grundlage von Milchinhaltsstoffen beschäftigten. Danach werden die in dieser Arbeit verwendeten Daten und Softwares vorgestellt. In Kapitel 4 werden zum einen die Grundlagen darüber dargestellt, welche Datenvorverarbeitungsschritte in einem Data Science Projekt allgemein durchgeführt werden, und welche konkret in dieser Arbeit angewendet wurden. Darauf folgen Erläuterungen über die angewendeten maschinellen Lernmethoden und die Funktionsweisen des Multi-Layer Perzeptrons, des Naive-Bayes-Klassifikators sowie der Support Vector Machine. In Kapitel 6 werden zum einen die Vorhersageergebnisse der Klassifikatoren dargestellt, zum Anderen wird erläutert, wie die Problemstellung angepasst werden muss, um die Anwendung von maschinellen Lernmethoden auf den Daten überhaupt zu ermöglichen. Zuletzt werden die Ergebnisse ausgewertet und mit ähnlichen Arbeiten verglichen. Den Abschluss bilden ein Fazit und ein Ausblick auf mögliche weiterführende Analysen.

2 Biologische Grundlagen

2.1 Laktation bei der Kuh

Damit Kühe Milch produzieren, müssen sie ein Kalb gebären. Der Zeitraum zwischen zwei Kalbungen wird als Laktationszyklus oder auch Laktationsphase bezeichnet. Dieser besteht aus vier Phasen: Der frühen, der mittleren und der späten Laktation sowie der Trockenphase. Während die Trockenphase zwischen 45-75 Tage anhält (Kuhn, Hutchison und Norman, 2007), dauern die die restlichen drei Phasen jeweils etwa 120 Tage. Während des gesamten Zeitraums verändert sich der Metabolismus der Kuh, um das Kalb bestmöglich versorgen zu können. Diese Veränderungen zeigen sich in der aufgenommen Menge an Trockenmasse, aber auch an der produzierten Milchmenge und dem Körpergewicht des Muttertiers. Die Variation dieser Parameter über den gesamten Laktationszyklus ist in Abbildung 2.1 dargestellt (Moran, 2009; Strucken, Laurenson und Brockmann, 2015).

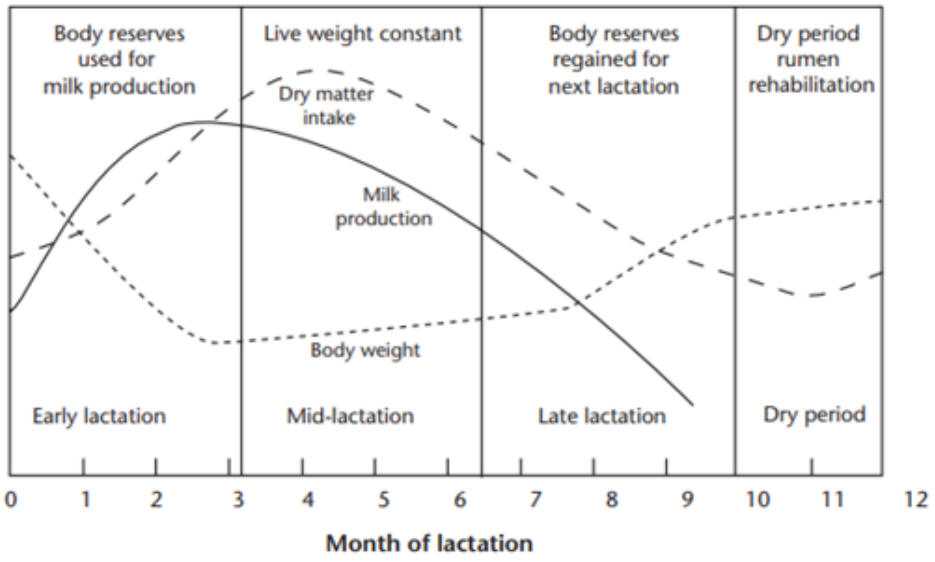


Abbildung 2.1: Schematischer Ablauf des Laktationszyklus einer Kuh und die damit einhergehenden Veränderungen in Milchproduktion, Körpergewicht und aufgenommener Trockenmasse. Aus Moran, 2009.

Die Milchproduktion pro Tag steigt nach der Geburt zunächst kontinuierlich an. Etwa 6 bis 12 Wochen nach dem Abkalben erreicht sie ihren Höhepunkt. Dieser Anstieg führt zwangsweise zu einem Energiedefizit des Muttertiers, gleichzeitig erreicht die pro Tag aufgenommene Trockenmasse erst in der Mitte der Laktation ihren Höhepunkt. Der Grund darin liegt darin im Pansen, welcher durch das wachsende Kalb verkleinert ist.

Um dem gesteigerten Energiebedarf durch die steigende Milchproduktion gerecht zu werden, nutzt die Kuh bis zu 12 Wochen nach dem Kalben neben der konsumierten Energie zusätzlich noch die eigenen Energiereserven. Aus diesem Grund muss bereits in der vorherigen Laktation bzw. Trockenphase ausreichend Körpermasse aufgebaut werden. Ist dies nicht der Fall, wird in der frühen Laktation die durch die Nahrung aufgenommene Energie dafür genutzt, die fehlende Körpermasse der Kuh aufzubauen (Moran, 2009). Diese Veränderungen des Metabolismus der Kuh können zu Erkrankungen führen und sich auf die Zusammensetzung der Milch auswirken. Wie genau sich einzelne Parameter verändern ist im folgenden Abschnitt erläutert.

2.2 Milchleistungsprüfung

Die Milchleistungsprüfung ist eine Untersuchung, die elfmal pro Jahr von unabhängigen Milchkontrollorganisationen bei laktierenden Kühen gemacht wird. Dabei werden Milchmenge und wichtige Milchinhaltsstoffe untersucht. Im Sachsen wird diese Untersuchung vom Sächsischen Landeskontrollverband (LKV) durchgeführt. Im Prüfjahr 2020/2021 wurden in Sachsen 546 Betriebe mit 164.901 Kühen (Stand September 2021) untersucht. Das macht eine Prüfdichte von 94,3 % im Freistaat aus (LKV Sachsen, 2022).
Aus den gesammelten Daten können Aussagen über die Herden- und Einzeltiergesundheit gemacht werden. Die wichtigsten Paramter, die im Rahmen der MLP analysiert werden, umfassen die Milchmenge jeder Kuh pro Tag, den Fett-, Eiweiß- sowie Laktoseanteil der Milch, die Harnstoffmenge, die Zellzahl sowie das Fettsäurespektrum. Außerdem wird dokumentiert, in welchem Laktationsstadium und Laktationstag sich eine Kuh zum Zeitpunkt der MLP befindet. Darüber hinaus sind aggregierte Daten enthalten, wie beispielsweise der Fett-Eiweiß-Quotient. Eine vollständige Übersicht aller Parameter ist in Tabelle A1 und A2 zu finden.

Aus den Milchinhaltsstoffen lassen sich Aussagen über die Versorgungssituation einer Kuh treffen, die Auswirkungen darauf hat, wie viel Milch eine Kuh pro Tag produzieren kann. Ein schematischer Verlauf der Milchinhaltsstoffe innerhalb einer Laktation ist in Abbildung 2.2 dargestellt. Maßgeblich ist die Menge an Energie, die die Kuh aus dem Futter zehren kann. Diese wird in Form von Kohlenhydraten, im Speziellen Stärke, Faserstoffen und Zucker aus dem Futter gewonnen und bei der Verdauung in Glukose umgewandelt. Diese Glukose wird im Anschluss im Eutergewebe zu Laktose synthetisiert. Da Laktose im Gegensatz zu anderen Milchinhaltsstoffen relativ konstant in der Milch enthalten sein muss, ist sie ein maßgebender limitierender Faktor für die zu bildende Milchmenge. Wenn eine Kuh nicht ausreichend Energie aus dem Futter gewinnt, werden ihre Stoffwechselprozesse auf eine alternative Glukosegewinnung umgestellt, um den Laktosegehalt in der Milch aufrecht zu erhalten. Diese Umstellung ist dann unter Umständen aus den MLP-Daten ablesbar (Glatz-Hoppe, Losand et al., 2020).
Eine Art, um auf alternativem Wege Energie zu gewinnen, stellt die Nutzung von Ami-

nosäuren dar. Diese sind bereits verdaut und befinden sich im Blutkreislauf und werden normalerweise für die Bildung von Milchprotein genutzt. Sobald diese Aminosäuren absorbiert werden, sind geringere Milcheiweißgehalte in der Milch zu finden (Glatz-Hoppe, Losand et al., 2020).
Ein weiterer Weg ist die Nutzung von körpereigenen Energiespeichern wie beispielsweise Fettreserven (vgl. Abb. 2.1). Bis zu einem bestimmten Grad ist dies ein unbedenklicher Regulationsmechanismus. Der verstärkte Abbau von Körperfett kann anhand erhöhter Milchfettgehalten festgestellt werden, denn die dabei entstehenden Stoffwechselprodukte werden in den Euter transportiert und bilden dort einen Teil des Milchfetts (Glatz-Hoppe, Losand et al., 2020; Moran, 2009).

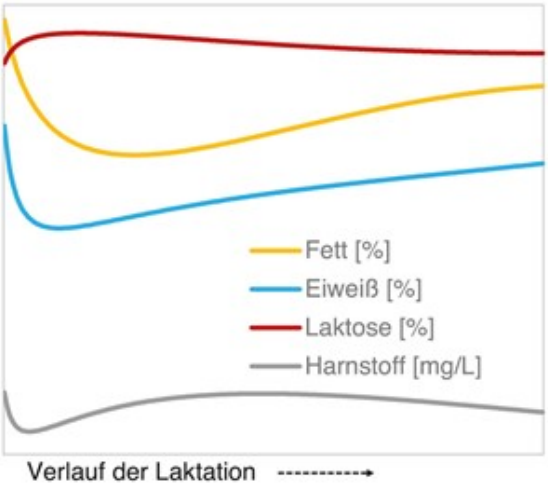


Abbildung 2.2: Veränderung ausgewählter Milchinhaltsstoffe im Verlauf der Laktation (Glatz-Hoppe, Losand et al., 2020).

Ein weiterer Parameter, der Auskunft über die Versorgung der Kuh gibt, ist der Milchharnstoffgehalt. Er korrelliert mit dem Harnstoffgehalt, der sich im Blut befindet, und drückt aus, wie viel des über das Futter aufgenommenen Eiweißes verwertet werden. Die Milchharnstoffgehalt hängt vorrangig von der aufgenommenen Rohproteinmenge und Proteinqualität ab, aber auch davon, wie viel dieses Rohproteins mikrobiell verarbeitet wurde. Diese Synthese findet im Pansen der Kuh statt, wodurch Rückschlüsse auf die mikrobielle Flora im Pansen möglich sind (Glatz-Hoppe, Mohr und Losand, 2019; Glatz-Hoppe, Losand et al., 2020).
Die Anzahl der Zellen in der Milch ist ebenfalls ein wichtiges Beurteilungskriterium für die Rohmilchqualität. Die Zellzahl beschreibt die Anzahl an somatischen (körpereigenen) Zellen pro Milliliter Milch und wird durch Vorgänge der Abwehrmechanismen beeinflusst. So kann eine hohe Zellzahl ein Indikator für Erkrankungen, insbesondere die des Euters, sein. Eutergesunde Tiere weisen einen durchschnittlichen Zellgehalt von weniger als 100.000 Zellen/ml Milch auf (Märtlbauer und Becker, 2016).

2.3 Literaturrecherche

Um einen ersten Überblick darüber zu gewinnen, inwiefern bereits Ansätze existieren, um mittels maschineller Lernverfahren anhand der Milchinhaltsstoffe möglicher Erkrankungen vorherzusagen, wurde zunächst eine Literaturrecherche durchgeführt. Dazu wurde die Literaturdatenbank *PubMed* des NCBI (National Center of Biotechnology Information) verwendet. Die nützlichsten Suchergebnisse wurden mit dem Suchbegriff „cow disease prediction milk“ erhalten, wobei sich darunter auch zahlreiche Ergebnisse befanden, die für die vorliegende Problemstellung nicht relevant waren. Um das Erkrankungsrisiko zu bestimmen, wurden häufig genetische bzw. transkriptomische Daten (Tzelos et al., 2022) oder Blutinhaltsstoffe (Lou et al., 2022) zusätzlich herangezogen. Außerdem kam es vor, dass in den Studien, die Milchinhaltsstoffe zur Vorhersage nutzten, andere Parameter oder Biomarker als die hier vorliegenden verwendet wurden. So wurden beispielsweise unter anderem der pH-Wert der Milch, der Kasein-Gehalt und ausführlichere Informationen über die Zahl und Zusammensetzung der somatischen Zellen in einer Arbeit genutzt, um den Gesundheitsstatus des Euters vorherzusagen (Bobbo et al., 2021). Insgesamt konnte keine wissenschaftliche Arbeit gefunden werden, in der die gleichen Daten für eine Vorhersage genutzt wurden, wie in unserem Fall geschehen soll. Dennoch soll im Folgenden auf zwei Paper eingegangen werden, die für die nachfolgenden Analysen hilfreich sein könnten.
Als am aufschlussreichsten scheint eine Systematic Literature Review von Slob, Catal und Kassahun, 2021 zu sein, die die Anwendungen des Maschinellen Lernens im Milchkuhsektor untersucht. Die Autoren analysierten anhand von 38 Primärquellen (von insgesamt 427 Veröffentlichungen), wie maschinelle Lernmethoden genutzt werden, um Vorhersagen über verschiedenste Variablen im Bezug auf die Milch der Kuh zu treffen. In mehr als der Hälfte der Fälle (55 %) wurde versucht, Krankheiten vorherzusagen oder zu erkennen. Daneben wurde sich auch mit der vorhersage von Milchausbeute (26%) und Milchqualität (19%) beschäftigt. Dafür wurden unter anderem Melkparameter und Milcheigenschaften (wie Leitfähigkeit, Farbe) mit einbezogen sowie Milchinhaltsstoffe, Informationen über die Kalbung, Schwangerschaft und Laktation, als auch Kuh- und Hofcharakteristiken. Am häufigsten wurden Decision Tree-basierte Algorithmen verwendet, gefolgt von Artificial Neural Networks. Regressionsbasierte Algorithmen und andere Methoden wurden lediglich in 13 Papern angewendet. Das liegt unter anderem daran, dass in 26 der Paper ein Klassifikationsproblem zu lösen war. Regressionsbasierte Algorithmen wurden dementsprechend vorrangig als Benchmark-Tests/Baseline-Performance-Tests genutzt. Bei der Krankheitserkennung wurden am häufigsten Decision-Tree-basierte Algorithmen verwendet. Um die Performanz der verwendeten Modelle zu bewerten, wurde bei fast zwei Dritteln der untersuchten Veröffentlichungen die Methoden der k-fold Cross-Validation eingesetzt. Die größten Probleme stellten die Feature Selection, d. h. die sinnvolle Auswahl der vorhandenen Features, und unbalancierte Datensätze dar. Bei unbalancierten Datensätzen liegt eine Klasse in unverhältnismäßig größerer Zahl als die andere Klasse vor, was das Trainieren des Modells erschwert. Darüber hinaus wurden Overfitting und Parameter Tuning als Probleme

identifiziert, sowie auch die allgemeine Größe des zu lösenden Problems, welche eine lange Verarbeitungszeit oder das Zerlegen in viele Teilschritte zur Folge hat.
Insgesamt kann diese Review erste Anhaltspunkte geben, welche Probleme bei der späteren Modellbildung auftreten können und welche maschinellen Lernmethoden und Evaluationsmöglichkeiten sinnvoll bzw. nicht sinnvoll sind.

Ein anderes Paper beschäftigte sich mit der Vorhersage des metabolischen Zustands von Kühen in den ersten sieben Wochen der Laktation anhand von Milchdaten. Dass der metabolische Zustand einer Kuh erheblich zu ihrem Gesundheitszustand beiträgt, ist bekannt. Aus diesem Grund ordneten Xu et al., 2019 334 Kühe anhand der Metabolite und Hormone, die ihrem Blut gefunden werden konnten, in drei Cluster ein: guter, mittlerer und schlechter metabolischer Zustand. Zu den Blutdaten wurden Milch- und Kuhdaten (Körpergewicht, Länge der Trockenperiode, Laktation, Milchausbeute, Fettgehalt, Fettanteil in %, Proteingehalt, Proteinanteil in %, Laktosegehalt, Laktoseanteil in %, Fett- und Protein-korriegerte Milch, somatische Zellzahl) erfasst, anhand derer dann acht maschinelle Lernmodelle entwickelt wurden (Decision Tree, Naive Bayes, Bayesian Network, Support Vector Machine, ANN (Artificial Neural Network), Bootstrap Aggregation, Random Forest und K-Nearest Neighbor). Hierbei wurde bereits eines der Probleme beobachtet, welche in der Review von Slob, Catal und Kassahun, 2021 angedeutet wurde. Es liegt eine stark ungleiche Klassenverteilung vor, sodass zu wenige Daten von Kühen mit einem schlechten metabolischen Zustand vorlagen. Aus diesem Grund wurden die Modelle nur auf die Vorhersage von 2 zwei Klassen trainiert. Mit Hilfe einer 10-fold Cross-Validation wurden dann die Modelle evaluiert. Dabei schnitten Random Forest und Support Vector Machine am besten ab. Außerdem wurde untersucht, wie stark welches der untersuchten Features zur Entscheidung beim Random Forest Modell beiträgt. In Woche 1 bis 3 spielten Milchmenge, Fettgehalt und Proteinanteil die größte Rolle, um zu entscheiden, welchem metabolischen Zustand eine Kuh zuzuordnen ist. In Woche 4 und 5 waren Proteinanteil in % und Laktosegehalt am ausschlaggebendsten. In Woche 6 und 7 waren sowohl Milchmenge, Proteinanteil, als auch Laktosegehalt wichtig, wohingegen der Fettanteil nicht mehr relevant war. Ob diese Ergebnisse sich in den vorliegenden Daten ebenso wiederfinden lassen, muss überprüft werden. Dass sich der Einfluss der verschiedenen Inhaltsstoffe in den ersten Wochen stark verändert, kann später starke Auswirkungen auf die Entwicklung und die Anpassung der Modelle haben.

3 Material

3.1 Datensätze

Die Grundlage für die Analyse bilden zwei Datensätze, welche vom LKS Labor Lichtenwalde zur Verfügung gestellt wurden (https://www.lkvsachsen.de/unternehmen/lks-mbh/). Ein Datensatz enthält die Daten der Milchleistungsprüfung, während der andere die Krankheitsdaten der jeweiligen Kühe enthält. Beide Datensätze beinhalten Kühe aus einem und dem demselben Betrieb.

3.1.1 Krankheitsdaten

Der Datensatz über die dokumentierten Erkrankungen der Kühe setzt sich insgesamt aus 14 Spalten mit 5619 Zeilen zusammen. Die erfassten Parameter sind in Tabelle A3 dargestellt. Der Datensatz enthält 411 verschiedene Ohrnummern, von denen sich 404 auch in dem Datensatz der Milchleistungsprüfungen wiederfinden. Die sechs Ohrnummern, die nicht in den MLP-Daten vertreten sind, sind folgende: 1404329024, 1404584716, 1404777870, 140477788, 1405066934, 1405858895. Außerdem befindet sich ein NA-Wert darunter, welcher für die nachfolgenden Analysen entfernt wurde.

3.1.2 MLP-Datensatz

Der Datensatz, der die Ergebnisse der Milchleistungskontrolle und die daraus abgeleiteten Ergebnisse enthält, besteht aus 39 Spalten mit insgesamt 4125 Beobachtungen. Die primären MLP-Daten, die direkt bei der MLP erfasst werden, sind in Tabelle A1 dargestellt. Die sekundären MLP-Daten, die aus den primären Daten errechnet werden, sind in Tabelle A2 zu sehen. Der MLP-Datensatz umfasst 413 Ohrnummern, von denen auch 404 in dem Erkrankungsdatensatz vertreten sind. Dementsprechend fehlen 9 Ohrnummern in den Krankheitsdaten. Dies sind folgende: 1404777929, 1405066941, 1404584735, 1404777882, 1405066958, 1405066924, 1405305688, 1405858854, 1405858848.

3.1.3 Zusammengeführter Datensatz

Um Vorhersagen über das Auftreten von Erkrankungen anhand der MLP-Daten zu ermöglichen, müssen die beiden Datensätze zusammengeführt werden. Das Vorgehen dafür ist in Abschnitt 4.2 dargelegt. Die Datensätze mit denen letztendlich gearbeitet

wurde, sind in 6.1 zu finden.

3.2 Software

Ein Großteil der Vorverarbeitung der Datensätze geschah mit der Programmiersprache R. Dies umfasst die Bereinigung der Daten, sowie die Feature Extraction und Selection. Als Entwicklungsumgebung wurde dabei RStudio genutzt. Konkret wurden R Version 4.1.2 und RStudio Version 1.3.1093 verwendet.
Neben der Standardbibliothek von R wurden auch drei zusätzliche Pakete genutzt, um die Daten zu untersuchen. Die Pakete *dplyr* und *tidyr* stellen Funktionen bereit, die die Vorverarbeitung der Daten ermöglichten, während das Paket *ggplot2* die Visualisierung der Daten in Form von Diagrammen ermöglichte. Dieses wurde insbesondere für die Darstellung der Ergebnisse der deskriptiven Statistik genutzt. Darüber hinaus wurde das Paket *lubridate* genutzt, um den Umgang mit Zeitangaben zu erleichtern.

Neben R wurde Python 3.9 verwendet. In dieser Programmiersprache erfolgte vorrangig die Anwendung der maschinellen Lernmodelle, als auch das Skalieren sowie die Balancierung der Daten. Die genutzte Entwicklungsumgebung ist PyCharm Community Edition V. 2022.1.2.
Um maschinelle Lernmethoden nicht von Grund auf neu programmieren zu müssen, wurde auf die Bibliotheken von *scikit-learn* V. 1.1.1. und *Keras* V. 2.9.0 zurückgegriffen. Letztere wurde genutzt, um das Multi-Layer Perzeptron zu implementieren, während *scikit-learn* den Einsatz von Support Vector Machines und Naive Bayes ermöglichte. Darüber hinaus wurden mit Hilfe jener Bibliothek eine Skalierung der Daten, Cross-Validation und die Berechnung von Wahrheitsmatrizen durchgeführt. Der Umgang mit csv-Dateien wurde durch *pandas* V. 1.4.2 möglich gemacht, während mit *numpy* mathematische Berechnungen erleichtert wurden. Die *seaborn*-Bibliothek (V. 0.11.2) wurde zusammen mit *matplotlib* (V. 3.5.2) für die Visualisierung der Ergebnisse genutzt.

Welche Methoden und Klassen konkret zum Einsatz kamen, wird an geeigneter Stelle in den Grundlagen der Datenvorverarbeitung (Kapitel 4) und des Maschinellen Lernens (Kapitel 5) erläutert. Sie werden jeweils mit **Eigenes Vorgehen** eingeleitet.

4 Datenvorverarbeitung

Um Daten für ein maschinelles Lernmodell nutzbar zu machen, müssen die erhobenen Daten an die jeweilige genutzte Methode angepasst werden. Den Vorgang, die Daten zu bereinigen, zu skalieren, und zu transformieren, wird als „Data Preprocessing“ (dt. *Datenvorverarbeitung*) bezeichnet. Dieser Schritt ist bei jedem Data Mining Prozess unerlässlich, da sonst der genutzte Algorithmus nicht funktionsfähig ist, oder kein akkurates Ergebnis liefert. Im Folgenden werden die wichtigsten Punkte, die bei der Vorverarbeitung der Daten durchgeführt werden müssen, erläutert.

4.1 Bereinigung

Bei der Bereinigung der Daten geht es darum, „schlechte“ Daten zu korrigieren, falsche Daten aus dem Datensatz herauszufiltern, und nicht nötige Details aus dem Datensatz zu entfernen. Häufige fehlerhafte Daten beinhalten fehlende Einträge, Schreibfehler, gemischte Formate, Duplikate oder Werte bzw. Einträge, die aufgrund von Regeln und Grenzen aus der realen Welt nicht möglich sein dürfen. Wie genau mit fehlenden Werten und Duplikaten umzugehen ist, unterscheidet sich von Projekt zu Projekt. Sie können entweder gelöscht werden oder als Einträge mit fehlenden Werten bzw. als Duplikate gekennzeichnet werden. Datensätze, die nicht in einem einheitlichen Format gehalten sind (z.B. bei Datumsangaben) können einheitlich gemacht werden, oder die Features, die in ihrem Format von den anderen abweichen, werden entfernt (Garcia, Luengo und Herrera, 2014).
Fehler in den Daten, die nicht formaler Natur sind, sondern inhaltlich fehlerhaft sind, können durch *qualitative* oder *quantitative* Techniken detektiert werden. Quantitative Techniken basieren auf statistischen Methoden, um abnormales Verhalten oder Ausreißer zu entdecken. Bspw. werden Werte, die mehr als drei Standardabweichungen vom Mittelwert entfernt sind, als Fehler angesehen. Quantitative Methoden hingegen nutzen Bedingungen, Regeln oder Muster, um Fehler zu erkennen. Bei Daten über den Arbeitsort, das Erfahrungslevel und das Gehalt von Arbeitnehmern, könnte zum Beispiel eine Regel lauten, dass es keine zwei Arbeitnehmern des selben Levels geben darf, bei denen das Gehalt desjenigen, der in München arbeitet, geringer ausfällt als das Gehalt desjenigen, der außerhalb von München arbeitet (Chu et al., 2016).
Sind solche Fehler gefunden worden, gilt es, sie zu beheben. Dafür gibt es verschiedene Ansätze, die in der Literatur beschrieben sind. Einige Beispiele lassen sich in Ilyas und Chu, 2019 oder Aggarwal, 2017 finden.

Eigenes Vorgehen:
Zunächst einmal wurden im MLP-Datensatz alle Datenpunkte entfernt, die irgendwo

einen NA-Werte enthielten. Bei dem Krankheitsdatensatz wurden nur die Datenpunkte entfernt, die in irgendeiner Spalte außer „Datum Abgang“, „Grund Abgang“, „Alter Abgang“, „Laktationstag Abgang“, „Diagnose“ oder „Klasse“ NA-Werte enthielten. Außerdem wurden jegliche Datumsangaben mit Hilfe des R-packages *lubridate* in ein vom Computer lesbares Format gebracht. Duplikate konnten keine festgestellt werden. Es wurden bei allen Analysen, wenn nicht anders angegeben, nur die Daten von Kühen verwendet, von denen sowohl MLP- als auch Krankheitsdaten vorhanden waren. Die Daten wurden nicht auf inhaltliche Fehler oder auf Ausreißer untersucht. Der Grund darin liegt im fehlenden Expertenwissen. Außerdem sollte vermieden werden, dass durch Outlier-Detection eventuell nützliche Informationen über das Auftreten von Erkrankungen verloren gehen.

4.2 Integration

Bei der Datenintegration geht es darum, Datensätze aus verschiedenen Quellen zusammenführen, sodass für die spätere Benutzung ein einziger Datensatz verwendet werden kann, der alle notwendigen Informationen enthält. Es ist insbesondere darauf zu achten, dass es bei der Datenintegration nicht zu Redundanzen und Inkohärenzen kommt (Garcia, Luengo und Herrera, 2014).

Eigenes Vorgehen:
In der vorliegenden Arbeit mussten die beiden Datensätze der MLP-Daten und der Krankheitsdaten zusammengeführt werden. Dies gestaltete sich sehr schwierig, da sich in den MLP-Daten keinerlei Information über die Laktationsnummer enthalten war. So musste anhand des Kalbedatums aus den MLP-Daten und dem Erkrankungsdatum der Krankheitsdaten ermittelt werden, in welcher Laktationsnummer eine MLP erfasst wurde. Ein großer Teil davon musste per Hand erfolgen. Der Grund dafür liegt darin, dass nicht in jeder Laktationsnummer, in der MLPs aufgenommen wurde, auch Krankheiten existierten, oder Krankheiten teilweise viel früher oder viel später als die MLPs auftraten. Letztendlich konnte aber fast allen MLP-Daten eine entsprechende Laktationsnummer zugeordnet werden. Dadurch konnte später die Problemstellung überprüft, weitere statistische Analysen durchgeführt als auch die jeweils auftretenden Krankheiten hinzugefügt werden. Der endgültige Datensatz ist in Kapitel 6.1 dargestellt.
Das Zusammenführen der Datensätze wurde in R V.4.1.2 mit den Paketen *dyplyr* und *tidyr* durchgeführt.

4.3 Skalierung

Daten müssen in der Regel normalisiert werden, bevor sie mit einem Algorithmus verarbeitet werden. Damit soll verhindert werden, dass Attribute mit großen numerischen Werten andere Attribute mit kleineren numerischen Werten „überschatten“ und dadurch

die Vorhersage verzerren. Insbesondere bei Lernmethoden, die auf Distanzen basieren, ist dies möglich. Stattdessen sollten alle Attribute in einem gegebenem Intervall dargestellt werden, oder anderweitig standardisiert werden. Dadurch sind sie untereinander vergleichbar. Es gibt verschiedene Methoden, von denen im Folgenden zwei vorgestellt werden.

4.3.1 Min-Max-Normalisierung

Bei der Min-Max-Normalisierung werden die Werte eines Attributs in einem bestimmten Intervall skaliert, nämlich $[new - min_A, new - max_A]$. Dabei stellen min_A und max_A das Minimum bzw. das Maximum der originalen Werte dieses Attributs dar. Konkret wird ein transformierter Wert v' so bestimmt:

$$v' = \frac{v - min_A}{max_A - min_A}(new - max_A - new - min_A) + new - min_A$$

In der Regel werden die Daten so normalisiert, dass das Intervall [0,1] bzw [-1,1] umfasst (Garcia, Luengo und Herrera, 2014).

4.3.2 Z-Score-Normalisierung

Eine Min-Max-Normalisierung ist nicht immer sinnvoll. Wenn bspw. Ausreißer („Outlier“) in den Daten vorliegen, können diese durch diese Art der Normalisierung nicht korrekt repräsentiert werden. Aus diesem Grund wird auf den sogenannten Z-Score zurückgegriffen. Dabei besitzen die Attributwerte nach der Normalisierung einen arithmetischen Mittelwert von 0 und eine Standardabweichung von 1. Der Z-Score z_i wird wie folgt berechnet:

$$z_i = \frac{x_i - \bar{x}}{s_x}$$

Dabei stellt x_i den zu standardisierenden Wert des i-ten Elements dar und $\bar{x}$ bzw. s_x bezeichnen den Mittelwert bzw. die Standardabweichung aller Elemente des betrachteten Attributs (Garcia, Luengo und Herrera, 2014). Der Z-Score kann außerdem genutzt werden, um Anomalien in den Daten zu identifizieren. Dafür muss ein Grenzwert festgelegt werden (Alam, 2020).

Eigenes Vorgehen:
Im vorliegenden Fall wurde der endgültige Datensatz, der nach der Integration entstanden ist, Z-skaliert. Dies geschah mit der *scikit-learn* Klasse *StandardScaler*.

4.4 Datendarstellung und abgeleitete Informationen

In diesem Schritt der Vorverarbeitung geht es darum, die vorliegenden Daten so zu konvertieren, dass der Data Mining Prozess effektiver wird oder gänzlich neue Attribute kreiert werden („Feature Construction“). Dazu gehört auch, qualitative bzw. kategorische Variablen in numerische Daten umzuwandeln, indem z.B. eine Binarisierung erfolgt (Garcia, Luengo und Herrera, 2014).

Eigenes Vorgehen:
Eine Binarisierung der vorliegenden Daten erfolgte bspw. vor der Multi-Label Klassifikation (siehe Abschnitt 5.1). Außerdem wurde im späteren Verlauf der Analysen versucht, zusätzliche Informationen über die vorherigen MLPs einzubeziehen und daraus neue Features zu erstellen. Der genaue Aufbau der Features wurde in Abschnitt 6.3.3 erläutert. All diese Anpassungen der Daten geschahen in R V.4.1.2 mit den Paketen *dplyr* und *tidyr*.

4.5 Feature Selection

Ziel der Feature Selection ist es, Features in dem Datensatz zu finden, die wichtig sind, und andere zu entfernen, die redundant oder irrelevant sind. Es soll also eine optimale Teilmenge der Features gefunden werden. Dadurch kann die Lerngeschwindigkeit der Algorithmen verbessert werden, die Komplexität des Modells verringert werden, oder auch die Vorhersagegenauigkeit erhöht werden. Bei Klassifikationsproblemen soll häufig die Teilmenge an Features gefunden werden, die die Vorhersagegenauigkeit maximiert. Dies kann mit verschiedenen Suchstrategien geschehen, welche in Garcia, Luengo und Herrera, 2014 beschrieben werden.

Eigenes Vorgehen:
Konkrete Suchstrategien wurden in dieser Arbeit nicht angewandt. Es wurden lediglich redundante oder irrelevante Features beim Trainieren des Modells ignoriert. Dazu gehören „Ohrnummer“, „Betrieb“ und die Features, die ein Datum enthalten. Letztere werden durch die Features „Laktationsnummer“ und „Laktationstag“ ausreichend repräsentiert. Erstere werden als irrelevant für die Vorhersage empfunden, da nur die Inhaltsstoffe der Milch beim Erstellen des Modells eine Rolle spielen sollten.

4.6 Instance Selection - Balancierung

Unter unbalancierten Daten versteht man einen Datensatz, bei dem es wesentlich mehr Einträge zu einer Klasse gibt, als zu einer anderen. Eine Klasse wird nur durch sehr

wenige Einträge repräsentiert („Minority Class“), während die andere Klasse den Großteil der Daten ausmacht („Majority Class“). Oft ist das bei Daten der Fall, die aus Anwendungen des realen Lebens entspringen, wie bspw. Daten über das Auftreten von Krankheiten, Detektion von Anomalien in Scans oder von Betrug bei Banknoten. Wenn eine Klasse die andere deutlich überwiegt, dann kann der Klassifikator zwar eine hohe Accuracy (näher erläutert in Abschnitt 5.3) aufweisen, bei genauerer Betrachtung wird allerdings nur die Majority Class korrekt klassifiziert - die Minority Class wird nur selten oder nie korrekt klassifiziert. Der Grund dafür liegt darin, dass die meisten Klassifikatoren versuchen die Error-Rate, also den Anteil der inkorrekten Vorhersagen des Klassenlabels, möglichst zu minimieren. Die Minority Class trägt oft wenig dazu bei, und es wird nicht unterschieden, wie kostenintensiv eine Falschklassifikation sein bzw. welche Auswirkungen sie haben kann (Ganganwar, 2012; KrishnaVeni und Sobha Rani, 2018).
Um zu umgehen, dass Modelle entstehen, die die Minority Class nicht berücksichtigen, gibt es Lösungen auf der Daten- und Algorithmusebene. Auf Ebene des Algorithmus können durch „Cost Sensitive Learning“ z.B. die Kosten der jeweiligen Klassen angepasst werden, sodass dem Ungleichgewicht entgegengewirkt wird. Auf Ebene der Daten werden „Sampling Methoden“ genutzt (Ganganwar, 2012).

4.6.1 Cost Sensitive Learning

Um das Problem der unbalancierten Daten bei Vorhersagen zu erläutern, soll das Beispiel der Tumorerkennung genutzt werden. Wenn ein Patient von einem Tumor betroffen ist, und dieser erkannt wird, handelt es sich um ein „True Positive“ Ergebnis. Ist der Patient nicht von einem Tumor betroffen, und es wird auch kein Tumor festgestellt, handelt es sich um ein „True Negative“ Ergebnis. Es fand keine Falschklassifikation statt, dementsprechend gibt es auch keine Strafkosten.
Wenn der Patient nun aber von einem Tumor betroffen ist, dieser allerdings nicht erkannt wird, handelt es sich um ein „False Negative“ Ergebnis. Wird ein positives Ergebnis für einen Patienten ohne Tumor vorhergesagt, handelt es sich um ein „False Positive“ Ergebnis. Diese Fälle sind Falschklassifikationen, und müssen bestraft werden. Jedoch sind die beiden Fälle nicht gleich schwerwiegend: Falsch-Negativ-Ergebnisse können unter Umständen zum Tod des Patienten führen, während bei einer Falsch-Positiven Klassifikation nur Kosten für einen weiteren Test aufkommen, der das Ergebnis bestätigt. Dementsprechend müssen die Kosten für eine Falsch-Negative Klassifikation wesentlich höher angesetzt werden, um die Situation realistisch widerzuspiegeln (KrishnaVeni und Sobha Rani, 2018).
Um die Kosten beim Cost Sensitive Learning darzustellen, wird eine Kostenmatrix genutzt (Abb. 4.1). Diese ähnelt in ihrem Aufbau einer einfachen Confusion Matrix (siehe Abschnitt 5.3), es sind allerdings statt der Anzahl der jeweiligen Ereignisse die Kosten für den jeweiligen Fall enthalten. Insgesamt soll ein Modell erstellt werden, bei dem die Gesamtkosten aus allen Klassifikationen möglichst gering ausfallen. Die Kosten für kor-

rekte Klassifikationen (True Positives und True Negatives) sind gleich 0, während die Kosten für Falschklassifikationen (False Positives und False Negatives) je nach Problemstellung angepasst werden. Im Falle der Tumorerkennung bedeutet das, dass die Kosten für eine Falsch Negativ Klassifikation größer sein müssen als für eine Falsch Positiv Klassifikation. Die Gesamtkosten für das Modell ergeben sich aus:

$$C(total) = C(FN) \cdot FN + C(FP) \cdot FP$$

(KrishnaVeni und Sobha Rani, 2018; McCarthy, Zabar und Weiss, 2005).

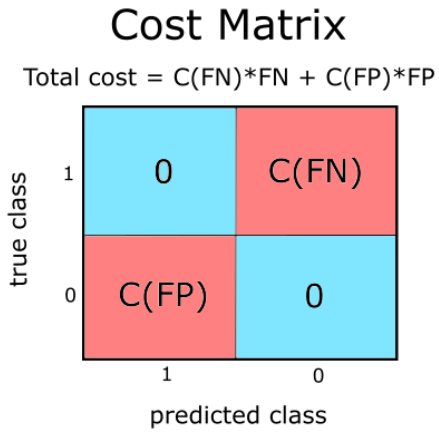


Abbildung 4.1: Allgemeine Darstellung einer Kostenmatrix. Adaptiert nach KrishnaVeni und Sobha Rani, 2018.

Eigenes Vorgehen
In der vorliegenden Arbeit wird kein Cost Sensitive Learning angewendet. Dementsprechend sind die Kosten aller möglichen Klassifikationsfälle gleich bzw. es existiert keine Kostenmatrix.

4.6.2 Sampling Methoden

Eine andere Variante, mit unbalancierten Daten umzugehen, sind sogenannte Sampling Methoden. Dabei wird der Datensatz neu angeordnet, damit die Klassen am Ende in etwa gleich stark vertreten sind. Diese Methoden gehören, im Gegensatz zum Cost Sensitive Learning, zur Preprocessing Phase und können dadurch bei jeder Lernmethode eingesetzt werden. Dadurch sollte auch jede Voreingenommenheit des Klassifikators gegenüber der Majority Class ausgeräumt werden (Ganganwar, 2012).
Allerdings sind Sampling Methoden nicht immer die sinnvollste Lösung. Abhängig vom Verhältnis von positiven zu negativen Datenpunkten, dem Aufbau des Datensatzes und der Funktionsweise des Klassifikators, werden beim Undersampling unter Umständen potentiell nützliche Daten entfernt, während beim Oversampling die Größe des Datensatzes erhöht und sich damit auch die Rechenzeit sowie das Rauschen in den Daten

erhöht (Van Hulse, Khoshgoftaar und Napolitano, 2007).
Im Folgenden soll auf die Unterschiede zwischen Over- und Undersampling als auch die unterschiedlichen algorithmischen Umsetzungen der beiden Methoden eingegangen werden.

Oversampling

Beim Oversampling wird die Minority Class vergrößert, sodass sie am Ende einen ähnlichen Umfang wie die Majority Class besitzt.
Die wohl einfachste Methode des Oversamplings ist das **Random Oversampling**. Dabei werden zufällig Daten aus der Minority Class dupliziert. Allerdings kann es durch das bloße Replizieren wahrscheinlicher zu Overfitting kommen.
Eine andere Methode zum Oversampling ist **SMOTE** („Synthetic Minority OverSampling Technique“). Dabei werden synthetische Datenpunkte erzeugt, indem zunächst von einem Datenpunkt der Minority Class die k nächsten Nachbarn bestimmt werden. Der Vektor aus einem dieser k nächsten Nachbarn und dem betrachteten Datenpunkt wird mit einer zufälligen Zahl x zwischen 0 und 1 multipliziert. Das Ergebnis wird zu dem betrachteten Datenpunkt addiert und bildet dann den neuen, synthetischen Datenpunkt (Chawla et al., 2002; Shelke, Deshmukh und Shandilya, 2017). Mit anderen Worten wird aus einem Datenpunkt der unterrepräsentierten Klasse mittels linearer Interpolation und mit Hilfe der k nächsten Nachbarn ein neuer Datenpunkt geschaffen. Problematisch ist dabei allerdings, dass bei dieser Technik neue Datenpunkte generiert werden, ohne die Majority Class zu berücksichtigen, Dadurch kann es zu Überlappungen zwischen Daten aus der Minority Class und Daten aus der Majority Class kommen, welches zu einer Übergeneralsierung der Modelle führen kann. Dennoch findet SMOTE durch seine Einfachheit häufig Anwendung und wurde vielfach erweitert und modifiziert (Ali et al., 2019).
Eine dritte Möglichkeit, den Umfang der unterrepräsentierten Klasse zu vergrößern, ist **ADASYN** („Adaptive Synthetic Sampling“). Dabei werden den Datenpunkten aus der Minority Class Gewichte zugeteilt, wenn in deren Umgebung mehr Datenpunkte aus der Majority Class zu finden sind, weil diese häufiger von Lernalgorithmen vernachlässigt werden (Ali et al., 2019). Dadurch werden die Entscheidungsgrenzen des Klassifikators verschoben und es werden Voreingenommenheiten, die durch die Verteilung der Klassen entstanden sind, verringert. Die Lernfähigkeit des Algorithmus verbessert sich (Tyagi und Mittal, 2020).

Undersampling

Beim Undersampling werden Datenpunkte der Majority Class entfernt. Insbesondere wenn es sich um große Datensätze handelt, wird diese Methode dem Oversampling oft vorgezogen (Tyagi und Mittal, 2020).

Random Undersampling ist die simpelste Methode, einen Datensatz zu balancieren. Dabei werden zufällige Datenpunkte der Majority Class gelöscht, bis der Umfang der beiden Klassen ähnlich ist. Problematisch bei dieser Methode ist jedoch, dass dadurch potentiell nützliche und informationsreiche Datenpunkte verloren gehen könnten (Ali et al., 2019).
Eine andere Möglichkeit, die überrepräsentierte Klasse zu reduzieren, ist der Einsatz von **Tomek Links**. Unter Tomek Links versteht man ein Datenpunkt-Paar, die zueinander die nächsten Nachbarn sind und zu unterschiedlichen Klassen gehören. Solche Paare sind in der Regel Grenzfälle bei der Klassifikation oder Rauschen, und werden deshalb oft falsch klassifiziert. Aus diesem Grund können solche Datenpunkte entfernt werden, ohne die Qualität des Datensatzes zu beeinflussen. Um Tomek Links bei der Balancierung eines Datensatzes zu nutzen, werden nur die Datenpunkte der Majority Class entfernt (Tyagi und Mittal, 2020).
Weitere Methoden des Undersamplings beinhalten eine Form der **k-nächste-Nachbarn-Klassifikation**, bei der das Klassenlabel eines Datenpunktes anhand seiner k nächsten Nachbarn bestimmt wird. Die Klasse, die in der Nachbarschaft am häufigsten vertreten ist, wird dem betrachteten Datenpunkt zugeordnet. Beim Undersampling wird der k-NN-Algorithmus in Verbindung mit Tomek-Links genutzt oder es wird ein 1-NN-Algorithmus verwendet, um Outlier zu detektieren (Tyagi und Mittal, 2020).

Eigenes Vorgehen:
In der vorliegenden Arbeit wurde Random Undersampling genutzt, weil diese Methode auch bei der Multi-Label Klassifikation anwendbar ist und somit alle Modelle untereinander vergleichbar gemacht werden. Es wurde dem Oversampling vorgezogen, da dies sonst zu zu viel Rauschen in den Daten geführt hätte.
Das Undersampling wurde mit Hilfe einer selbst geschriebenen Funktion in Python erreicht, welche die Zahl der MLPs, nach denen eine Erkrankung auftrat, auf die nächsten Hunderter aufrundet und so viele „gesunde“ MLPs zufällig auswählt. Dadurch ist das Klassenverhältnis ungefähr gleich, ohne die Überlegenheit der gesunden Klasse vollständig zu vernachlässigen.

5 Anwendung Maschineller Lernverfahren

Heutzutage wird eine große Menge an Daten erzeugt, sodass deren Auswertung nur noch mit Algorithmen und mathematischen Modellen möglich ist. Eine wichtige Rolle spielt dabei das Maschinelle Lernen. Darunter wird das automatisierte Auffinden von Regelmäßigkeiten in Daten mit Hilfe von Computeralgorithmen verstanden.
Im Gegensatz zu klassischen statistischen Methoden wird beim Maschinellen Lernen nicht von einer Beobachtung auf die Population geschlussfolgert, sondern es werden generalisierbare vorhersagbare Muster gesucht. Es soll also nicht mathematisch dargestellt werden, wie bspw. ein biologisches System funktioniert, sondern es sollen noch unbekannte Resultate bzw. zukünftiges Verhalten vorhergesagt werden können. Dazu müssen nur minimale Annahmen über die Daten-generierenden Systeme getroffen werden und es können Vorhersagen erzielt werden, selbst wenn die Daten ohne genau kontrolliertes Experimentdesign erhoben wurden. Bei statistischen Modellen ist eine kontrollierte Erhebung der Daten unerlässlich, um später wahrheitsgemäße Schlussfolgerungen über Zusammenhänge im System treffen zu können. Das Problem beim Maschinellen Lernen liegt darin, dass unter Umständen nicht nachvollzogen werden kann, welche internen Zusammenhänge zu einer Vorhersage geführt haben, auch wenn diese überzeugend ist (Bzdok, Altman und Krzywinski, 2018).

Zwei große Teilgebiete des Maschinellen Lernens stellen überwachte Lernverfahren und nicht überwachte Lernverfahren dar. Beim überwachten Lernen sollen Zusammenhänge zwischen Eingabevariablen und Zielvariablen erkannt werden, sodass nach erfolgreichem Lernen die Zielvariable bzw. das Klassenlabel nur anhand der Eingabevariablen vorhergesagt werden kann. Diese Zusammenhänge werden in einem Modell repräsentiert. Ein Modell beschreibt versteckte Beziehungen und Muster innerhalb der Daten, die die Zugehörigkeit zu einer Klasse erklären (Garcia, Luengo und Herrera, 2014; Rokach und Maimon, 2007).
Die zwei grundlegenden Probleme, die zum Überwachten Lernen gehören, sind Klassifikation und Regression. Bei der Klassifikation ist die Zielvariable in ihrem Umfang finit und kategorisch und wird deswegen auch oft als „Klassenlabel“ bezeichnet. Der Klassifikator muss also nach dem Training einem noch unbekannten und ungelabelten Datenpunkt ein Klassenlabel zuordnen. Im Gegensatz dazu muss bei einem Regressionsproblem eine infinite Zielvariable vorhergesagt werden. Die Zielvariable wird dabei als Funktion der Eingabevariablen verstanden, und das Modell muss daran angepasst werden. Aus diesem Grund sind Regressionsprobleme häufig komplexer und mit mehr Rechenaufwand verbunden (Garcia, Luengo und Herrera, 2014).
Im Gegensatz dazu sind beim unüberwachten Lernen nur Eingabevariablen vorhanden, es gibt keine Zielvariablen die vorhergesagt werden sollen. Ziel ist es hier, Regelmäßigkeiten, Beziehungen oder Ähnlichkeiten der Eingabevariablen zu finden. Das Modell

bindet auf Grundlage dieser selbstständig Cluster bzw Gruppierungen, denen die Daten zugehörig sind (Garcia, Luengo und Herrera, 2014).
In der vorliegenden Arbeit wurden ausschließlich Klassifikationsmethoden angewendet, was sich durch die Problemstellung begründet.

5.1 Klassifikation

Wenn Dateneinträge vordefinierten Klassen $C_1, ..., C_i$ zugeordnet werden sollen, kann die Klassifikation in eine der folgenden Aufgaben fallen (Sokolova und Lapalme, 2009):

5.1.1 Binäre Klassifikation

Bei der binären Klassifikation soll die Eingabe in eine von zwei Klassen (C_1, C_2) klassifiziert werden. Dies ist zum Beispiel der Fall, wenn ein Testergebnis vorhergesagt werden soll (positiv oder negativ). Dabei handelt es sich um die populärste Art der Klassifikation.

5.1.2 Multi-Class Klassifikation

Hierbei soll die Eingabe einer von i Klassen C_i zugeordnet werden. Dies ist bspw. beim Iris-Datensatz der Fall, wenn eine der drei Iris-Klassen *Iris setosa, Iris versicolor* oder *Iris virginica* vorhergesagt werden soll (Duda, 1973).

5.1.3 Multi-Label Klassifikation

Die Eingabe wird in mehrere von i sich überschneidenden Klassen C_i klassifiziert. Das ist später in der Arbeit der Fall, wenn versucht wird, die verschiedenen Erkrankungsarten vorherzusagen (Abschnitt 6.3.1).
Binäre, Multi-Class und Multi-Label Probleme gehören zur sogenannten „flachen Klassifikation“, bei der die Klassen separiert voneinander sind und keine Beziehung zwischen ihnen angenommen wird (Yang, 1999).

5.1.4 Hierarchische Klassifikation

Bei der hierarchischen Klassifikation werden die Beziehungen zwischen den Klassen berücksichtigt und deren Struktur wird in den Zielvariablen einbezogen. Die Eingabe wird in eine Klasse C_i eingeordnet, welche in Subklassen unterteilt oder in übergeordnete Klassen gruppiert ist. Die Hierarchie ist festgelegt und darf während der Klassifikation nicht verändert werden. Diese Klassifikationsaufgabe wird häufig in der Bioinformatik

verwendet, zum Beispiel bei der Vorhersage von Proteinfunktionen (Eisner et al., 2005).

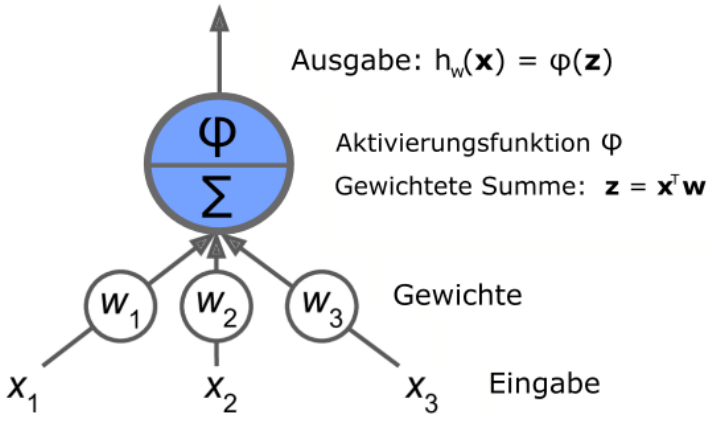


Abbildung 5.1: Schematischer Aufbau eines Perzeptrons. Übernommen und verändert aus Géron, 2019.

5.2 Klassische Modelle

5.2.1 Multi-Layer Perzeptron

Das Multi-Layer Perzeptron ist eine spezielle Form eines Künstlichen Neuronalen Netzwerks, deren Strukturen inspiriert von der Funktionsweise des Gehirns sind. Das Gehirn besteht aus einzelnen Neuronen, die miteinander verbunden sind und untereinander Signale weitergeben sobald ein bestimmtes Aktivierungspotential überschritten ist. Dieser Prozess wurde 1943 von McCulloch und Pitts, 1943 abstrahiert und als einfaches Modell vorgestellt, welches später als *künstliches Neuron* bekannt wurde. Diese künstlichen Neuronen können logische Funktionen wie AND, OR oder NOT durchführen.
Aus diesem Ansatz erwuchs das sogenannte *Perzeptron*, welches von Rosenblatt, 1958 entwickelt wurde. Es basiert auf der sogenannten *threshhold logic unit* (TLU), welche aufgebaut ist wie ein künstliches Neuron und über Zahlen als Ein- und Ausgabe verfügt. Wie in Abb. 5.1 dargestellt ist, erhält die TLU n Inputs über jeweils eine gewichtete Verbindung. Die gewichtete Summe ($z = w_1x_1 + w_2x_2 + ... + w_nx_n = \mathbf{x}^T\mathbf{w}$) aller erhaltenen Eingaben wird berechnet, eine Aktivierungsfunktion ϕ auf diese Summe angewendet und eine Ausgabe erzeugt ($h_w(\mathbf{x}) = \phi(z)$, wobei $z = \mathbf{x}^T\mathbf{w}$). Es existieren verschiedene Aktivierungsfunktionen, von denen eine Auswahl in Abb. 5.2 zu sehen sind (Géron, 2019; Zhou, 2021).

Eine einzelne TLU kann genutzt werden, um einfache lineare binäre Klassifikation durchzuführen. Es wird eine Linearkombination aller Eingaben gebildet und gegen einen Grenzwert verglichen. Übersteigt das Ergebnis den Grenzwert, wird die positive Klasse als Ergebnis ausgegeben, ansonsten die negative Klasse. Die TLU wird dann oftmals

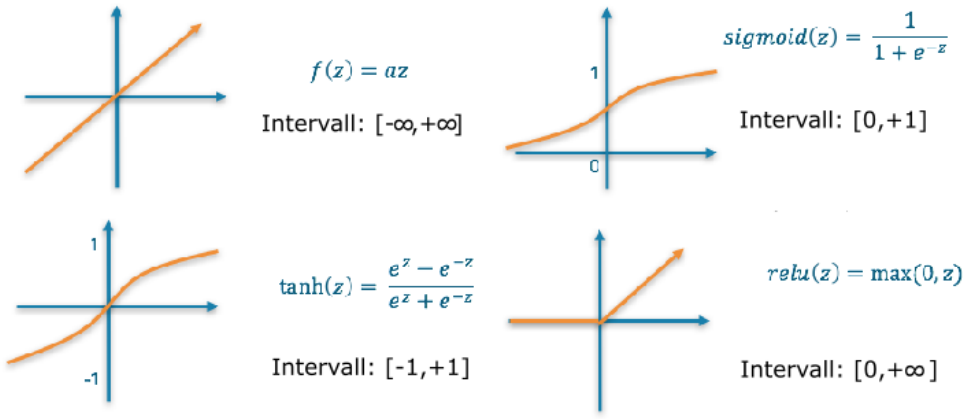


Abbildung 5.2: Beispiele einiger Aktivierungsfunktionen. Übernommen und verändert aus Ronaghan, 2018.

als *Perzeptron* bezeichnet, auch wenn unter diesem Begriff auch ein ganzer Layer von TLUs verstanden werden kann. In so einem Fall können mehr als zwei Klassen klassifiziert werden. Ein Perzeptron enthält auch immer ein sogenanntes *Bias-Neuron*, dessen Ausgabe entweder konstant bleibt oder angepasst werden kann (Géron, 2019).
Die Ausgabe eines Perzeptrons oder mehrerer verbundener Perzeptrone kann mit Gleichung 5.1 wie folgt berechnet werden:

$$h_{\mathbf{W},\mathbf{b}}(X) = \phi(\mathbf{XW} + b) \tag{5.1}$$

X und **W** repräsentieren dabei die Matrix der Eingabefeatures bzw. die der Gewichte der Verbindungen (außer die der Bias Neurone). Der Bias Vektor **b** enthält alle Gewichte der Verbindungen zwischen dem Bias Neuron und den TLUs. ϕ ist die Aktivierungsfunktion (Géron, 2019).
Trainiert wird das Perzeptron mit der *Hebb'schen Lernregel*, die besagt, dass das verbindende Gewicht zwischen zwei Neuronen erhöht wird, wenn beide Neuronen dieselbe Ausgabe liefern. Im Perzeptron bedeutet das, dass Verbindungen gestärkt werden, die dabei helfen den Fehler, den das Netzwerk bei der Klassifikation macht, zu reduzieren. Dem Netzwerk wird konkret ein Trainingsbeispiel nach dem anderen zugeführt und die Vorhersage des Netzwerks mit dem tatsächlichen Klassenlabel verglichen. Für jedes Ausgabeneuron, welches eine falsche Vorhersage lieferte, werden die Gewichte verstärkt, deren Eingaben zu einer korrekten Vorhersage beigetragen hätten. Die Regel ist in Gleichung 5.2 zu sehen.

$$w_{i,j}(\text{next step}) = w_{i,j} + \eta(y_j - \hat{y}_j)x_i \tag{5.2}$$

$w_{i,j}$ ist das verbindende Gewicht zwischen dem i-ten Eingabe-Neuron und dem j-sten

Ausgabe-Neuron. x_i beschreibt den i-ten Eingabewert des aktuellen Trainingsdatenpunkts. $\hat{y}_j$ ist die Ausgabe des j-sten Ausgabe-Neurons, während y_i die eigentliche Ausgabe dieses Neurons darstellt. η beschreibt die Lernrate (Géron, 2019).
Um Probleme zu lösen, die von komplexerer Natur als ein binäres Klassifikationsproblem sind, können mehrere Perzeptrone hintereinander geschalten werden. Solch eine Struktur wird als *Multi-Layer Perzeptron* (kurz: MLP, nicht zu verwechseln mit MLP - Milchleistungsprüfung) bezeichnet (Minsky und Papert, 1969). Es besteht aus einem Input Layer, einem oder mehreren Hidden Layers und einem Output Layer. Der Input-Layer besteht aus Input- oder Eingabeneuronen, welche die erhaltene Eingabe lediglich weiterleiten. Ein Eingabeneuron ist mit jedem Neuron aus dem Hidden Layer verbunden. Die Neurone der Hidden Layer und die des Output-Layers verfügen über Aktivierungsfunktionen, die sich pro Layer unterscheiden können. Jedem Layer außer dem Input Layer ist außerdem ein Bias Neuron zugehörig. Ein beispielhaftes Multi-Layer Perzeptron ist in Abb. 5.3 dargestellt. Dabei handelt es sich um ein *feedforward* Netz, bei dem die Signale nur in eine Richtung geleitet werden, nämlich vom Input Layer zum Output Layer. Bei dieser Architektur findet allerdings kein Anpassen der Gewichte und des Bias statt (Géron, 2019).

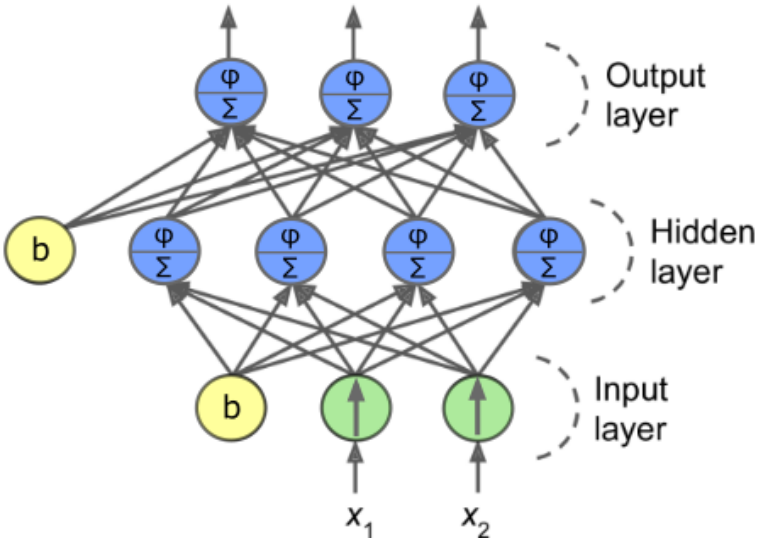


Abbildung 5.3: Schematische Darstellung eines Multi-Layer Perzeptrons. b repräsentiert den Bias. Übernommen und verändert aus Géron, 2019.

Zum Training des Multi-Layer Perzeptrons kommt es erst beim *backpropagation learning*, welches erstmals von Rumelhart, Hinton und Williams, 1986 vorgestellt wurde. Dabei handelt es sich um einen Algorithmus, der mit Hilfe von *Gradient Descent* bestimmt, wie stark jedes Gewicht und jeder Bias angepasst werden muss, um den Fehler in der Vorhersage anzupassen. Heutzutage wird allerdings häufiger auf den **Stochastic Gradient Descent** zurückgegriffen, da dieser schneller konvergiert.
Zuerst wird anhand der Daten des Trainingsdatensatzes eine Vorhersage durch das Netzwerk gemacht. Die Eingabe wird also vom Input-Layer bis zum Output-Layer durch das Netz propagiert, bis eine Ausgabe erhalten wird („Forward Propagation“). Diese

Ausgabe des Netzes wird über eine Loss Function mit der tatsächlich gewünschten Ausgabe verglichen. Bei der Loss Function handelt es sich um eine zuvor gewählte Funktion, die abhängig von der Problemstellung, den Unterschied zwischen vorhergesagtem Label und tatsächlichem Label evaluiert. Es existieren verschiedene Loss Functions, oder auch Cost Functions, welche in Tabelle 5.1 dargestellt sind. Der Wert der Loss Function, also der Fehler der Vorhersage, soll vom Algorithmus minimiert werden (Géron, 2019).

Loss Functions	
Mean Squared Error (MSE)	$= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
Cross Entropy	$= -\sum_{i}^{M} y_i \log(\hat{y}_i)$
Binary Cross Entropy	$= -(y\log(\hat{y}) + (1-y)log(1-\hat{y}))$

Tabelle 5.1: Beispiele einiger Loss Functions und deren Berechnung. $\hat{y}$ beschreibt den vorhergesagten Wert, während y den tatsächlichen Wert darstellt. M steht für die Anzahl der Klassen.

Um den Fehler der Vorhersage zu reduzieren und die Gewichte und den Bias aller Neuronen anzupassen, wird nach der Forward Propagation der Gradient als multi-variable Ableitung der Loss Function mit Rücksicht auf alle Netzwerk-Parameter berechnet. Grafisch gesprochen handelt es sich dabei um den Anstieg der Tangente der Loss Function im aktuellen Punkt (also bei den aktuell betrachteten Parametern). Da versucht wird, die Loss Function zu minimieren, muss entgegen des Gradienten gegangen werden. Um herauszufinden, welchen Einfluss das einzelne Neuron mit seinen Parametern auf den Fehler hat, wird rückwärts durch das Netz propagiert („Backward Propagation“). Zunächst werden die partiellen Ableitungen der Loss Function in Bezug auf alle Parameter des Output-Layers bestimmt, da dieser keinen Einfluss auf andere Netzwerk Parameter hat. Dies ist dank der Kettenregel schnell und unkompliziert. Danach werden die partiellen Ableitungen der Loss Function des vorhergehenden Layers bestimmt, was durch die soeben errechneten Ableitungen des Output-Layers ermöglicht wird. Dieser Vorgang wird weitergeführt, bis der Input-Layer erreicht wird (Géron, 2019).
Sobald der Gradientenvektor bestimmt wurde, werden die Gewichte des Netzwerks aktualisiert, indem der korrespondierende Gradientenwert der Loss Function $f(w,x)$ im Hinblick auf die Gewichte ($\nabla_w f(\mathbf{w,x})$) vom aktuellen Gewicht **w** subtrahiert und mit einer Lernrate η multipliziert wird. Dieser Prozess wird wie folgt geschrieben:

$$\mathbf{w} == \mathbf{w} - \eta \nabla_w f(\mathbf{w,x}) \tag{5.3}$$

Dieser Vorgang wird wiederholt, bis es zur Konvergenz kommt.

Gradient Descent besitzt allerdings einige Nachteile. Es kann sehr lange dauern, bis der Algorithmus konvergiert, und es kann passieren, dass der Algorithmus in ein loka-

les Minimum gerät, welches nicht die optimale Lösung darstellt (Zhou, 2021). Um diese Probleme zu beheben, wurde eine Vielzahl von Optimierungsalgorithmen entwickelt, von denen im Folgenden eine kleine Auswahl vorgestellt werden soll.

Bei der **Momentum Optimierung** besteht das Grundprinzip darin, in Richtung des Minimums immer mehr zu beschleunigen und verringert somit die Rechenzeit (Polyak, 1964). Während beim Gradient Descent die vorherigen Gradienten nicht einbezogen werden, werden sie bei der Momentum Optimierung berücksichtigt. Bei jedem Durchlauf wird der lokale Gradient vom Momentum-Vektor m subtrahiert und aktualisiert die Gewichte, indem ebenjener Moment-Vektor addiert wird. Der Gradient wird also zur Beschleunigung genutzt. Außerdem wird ein weiterer Parameter β, das Momentum, eingeführt, welches zwischen 0 und 1 liegt. Es beschreibt eine Art Reibung, die verhindert, dass der Algorithmus beim Erreichen des Minimums stark oszilliert und sich stattdessen schnell stabilisiert. Je höher der Wert, desto weniger Reibung wird eingebracht. Ein typischer Wert ist 0,9. Der Algorithmus ist in Gleichung 5.4 dargestellt (Géron, 2019).

$$\begin{aligned} \mathbf{m} &\leftarrow \beta\mathbf{m} - \eta\nabla_w f(\mathbf{w},\mathbf{x}) \\ \mathbf{w} &\leftarrow \mathbf{w} + \mathbf{m} \end{aligned} \tag{5.4}$$

Der Momentum Algorithmus konvergiert schneller als der Gradient Descent Algorithmus und kann dabei helfen, lokale Minima zu überwinden.
Andere optimierte Alternativen zum einfachen Gradient Descent sind *AdaGrad* (Duchi, Hazan und Singer, 2011) oder *Adam* (Kingma und Ba, 2014).

Eigenes Vorgehen:
Es wurden verschiedene Multi-Layer Perzeptrone angewendet, die je nach Klassifikationsproblem eine andere Struktur aufweisen. Alle Multi-Layer Perzeptrone bestehen aus einem Input Layer, einem Hidden Layer und einem Output-Layer. Der Input-Layer besitzt 36 Eingabe-Neuronen, so viele, wie Features vorhanden sind. Die genauen Features können in Abschnitt 6.3 nachgelesen werden. Der Hidden-Layer umfasst 5 Neurone, bei deren Aktivierungsfunktion es sich um die ReLu-Funktion handelt (dargestellt in Abschnitt 5.2.1). Als Loss Function wurde einheitlich die Binary-Cross-Entropy verwendet. Als Optimizer wurde der Adam-Algorithmus verwendet.
Um Multi-Label Klassifikationsprobleme zu lösen, wurden dem Output-Layer so viele Ausgabe-Neuronen zugewiesen, wie es Klassenlabel gibt. Wenn versucht wird, alle Erkrankungen vorherzusagen, sind es 10 Ausgabe-Neuronen. Wenn nur die relevanten Erkrankungen betrachtet werden, existieren 7 Ausgabe-Neuronen. Bei deren Aktivierungsfunktion handelt es sich um die Sigmoid-Funktion. Binary-Cross-Entropy kann hier als Loss-Function genutzt werden, da ein Multi-Label Klassifikationsproblem auch als ein Problem bestehend aus mehreren einzelnen binären Klassifikationen aufgefasst

werden kann. Die Vorhersage der jeweiligen Klassenlabels wird mittels der binären Cross-Entropy bewertet, und am Ende werden die Losses aller Label für die jeweilige Ausgabe zusammengefasst (Martinek, 2020).
So ähnlich sieht der Aufbau des Multi-Layer Perzeptrons auch bei einer binären Klassifikation aus. Da in diesem Fall nur ein Klassenlabel existiert, welches entweder den Wert 0 oder 1 annehmen kann, gibt es auch nur ein Ausgabe-Neuron.
Die Multi-Layer Perzeptrons wurden mit Hilfe der *Keras deep learning library* V. 2.9.0 erstellt (Chollet et al., 2015). Konkret wurde das Modell mit Hilfe der Klasse *Sequential* kreiert.

5.2.2 Support Vector Machine

Eine Support Vector Machine ist ein geometrischer Klassifikator. Das Funktionsprinzip basiert darauf, dass Daten geclustert und gruppiert werden können. Die Trainingsdaten werden durch jeweils einen Vektor in einem Vektorraum repräsentiert. Ziel der Support Vector Machine ist es, die Trainingsdaten je nach ihrer zugehörigen Klasse möglichst linear voneinander zu trennen. Teilweise ist das beim Auftragen der Punkte gegen ein Koordinatensystem schon deutlich erkennbar, wie in Abbildung 5.4 A zu sehen ist. Bei zweidimensionalen Systemen ist das durch eine Gerade möglich, bei mehreren Dimensionen durch eine sogenannte Hyperebene. Bei der Auswahl der Hyperebene soll ein möglichst großer Abstand zu den nächstgelegen Vektoren gewonnen werden. Diese am nächsten zur Hyperebene gelegenen Vektoren werden als „Stützvektoren“ (engl. Support Vectors) bezeichnet. Nur diese sind nötig, um die Ebene mathematisch exakt zu beschreiben. Durch den zu maximierenden Abstand zwischen Stützvektoren und Hyperebene wird dafür gesorgt, dass auch unbekannte Objekte möglichst zuverlässig klassifiziert werden. Die Support Vector Machine wird deswegen auch als „Large Margin Classifier“ bezeichnet (Noble, 2006).

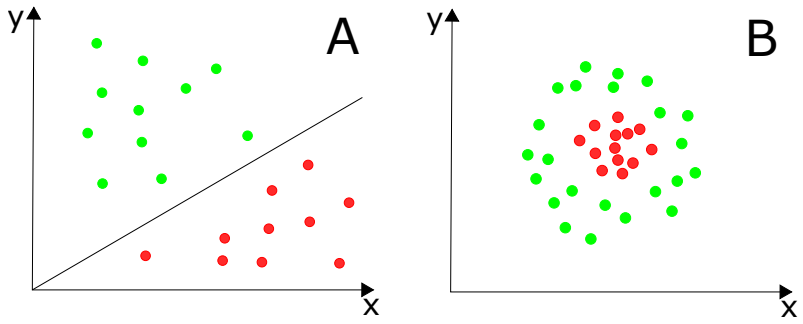


Abbildung 5.4: Trennbarkeit von Datenpunkten. In Abbildung A sind die Datenpunkte linear voneinander trennbar, in Abbildung B nicht. Die Abbildungen wurden mit Inkscape V. 1.0 erstellt.

Obwohl die SVM einzelne Abweichungen erlaubt und einzelne Objekte falsch klassifiziert werden dürfen, sind Daten manchmal dennoch schwer linear trennbar, wie in Abbildung 5.4 B verdeutlicht ist. Eine Lösung hierfür bietet die sogenannte Kernel-Funktion.

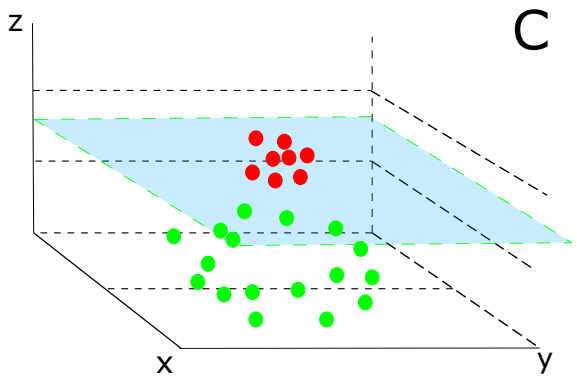


Abbildung 5.5: In dieser Abbildung wird das implizite nicht-lineare Mapping der Daten explizit visualisiert. In der expliziten Darstellung können die zuvor nicht trennbaren Datenpunkte durch eine Hyperebene (blau) separiert werden. Die Abbildung wurde mit Inkscape V. 1.0 erstellt.

Dies ist ein mathematischer Trick, der Daten implizit aus einem niedrig-dimensionalen Bereich in eine höhere Dimension umwandelt, wodurch eine lineare Trennung der Klassen ermöglicht wird. Das wird in Abb. 5.5 veranschaulicht. Dort wurden die Expressionswerte quadriert und die Datenpunkte können nun durch eine Hyperebene separiert werden (Noble, 2006).

Durch das implizite nicht-lineare Mapping wird eine lineare Trennung der Daten oft ermöglicht. Allerdings sollte die Verwendung von hochdimensionalen Kernel-Funktionen vermieden werden, da es so zu einer Überanpassung an den Trainingsdatensatz kommen kann. Das bedeutet, dass das Modell die Trainingsdaten „auswendig“ lernt und später an den Testdaten keine korrekte Klassifikation vornehmen kann (Noble, 2006). Ein Hinweis darauf kann sein, wenn viele Trainingsdatenpunkte zu Stützvektoren werden.

Die wohl wichtigsten Parameter sind der gewählte Kernel, der Regulierungsparameter C, sowie der Kernelkoeffizient γ. Deren Bedeutung wird im Folgenden erläutert.
Herauszufinden welche Kernel-Funktion für welchen Datensatz am geeignetsten ist, ist nur durch Ausprobieren möglich. Aus diesem Grund wurden in der vorliegenden Arbeit ein linearer, ein polynomialer Kernel, ein RBF-Kernel und ein Sigmoid-Kernel verwendet. Deren mathematische Formulierung lautet wie folgt:

Es gilt:$\langle \mathbf{x},\mathbf{x}' \rangle = \mathbf{x}^T\mathbf{x}$:

- linear: $\langle \mathbf{x},\mathbf{x}' \rangle$
- polynomial: $(\gamma\langle \mathbf{x},\mathbf{x}' \rangle + r)^d$
- rbf: $exp(-\gamma\|\mathbf{x}\text{-}\mathbf{x}'\|^2)$

- sigmoid: $tanh(\gamma\langle \mathbf{x},\mathbf{x}'\rangle + r)$

Der Parameter γ gibt an, wie viel Einfluss ein einziges Trainingsbeispiel auf die Entscheidung hat, wie die beiden Klassen voneinander zu trennen sind. Ein niedriger γ-Wert sorgt dafür, dass das Modell generalisierender wird und viele Punkte, die auch in einem größeren Radius entfernt voneinander liegen können, zu einer Klasse gezählt werden. Ein hoher γ-Wert hingegen kann dazu führen, dass das Modell zu spezifisch wird und es zur Überanpassung kommt. Die Datenpunkte müssen sehr nah beieinander liegen, um zu einer Klasse gezählt zu werden. Wird γ zu hoch gewählt, umfasst der Einflussbereich unter Umständen nur noch den Stützvektor selbst. Wird γ allerdings zu niedrig gewählt, kann es dazu kommen, dass das Modell trotz einer Kernel-Funktion sich wie ein linearer Klassifikator verhält, der lediglich Zentren mit hoher Dichte zweier Klassen voneinander trennt (Yıldırım, 2020; Pedregosa et al., 2011b).
Der zweite zu beachtende Parameter beim Einsatz einer SVM ist der Parameter C. Mit ihm wird versucht, einen Kompromiss zwischen einer möglichst geringen Zahl an Missklassifikationen und einem möglichst großen Abstand der Hyperebene zu den Stützvektoren zu finden. Je größer C gewählt wird, desto kleiner darf der Abstand zu Hyperebene werden, wenn dabei weniger Falschklassifikationen auftreten. Wenn C kleiner ist, wird ein größerer Abstand und damit auch eine einfachere Entscheidungsfunktion erreicht. Dies geschieht zu Ungunsten der Vorhersagegenauigkeit (Yıldırım, 2020; Pedregosa et al., 2011b; Duan, Keerthi und Poo, 2003).

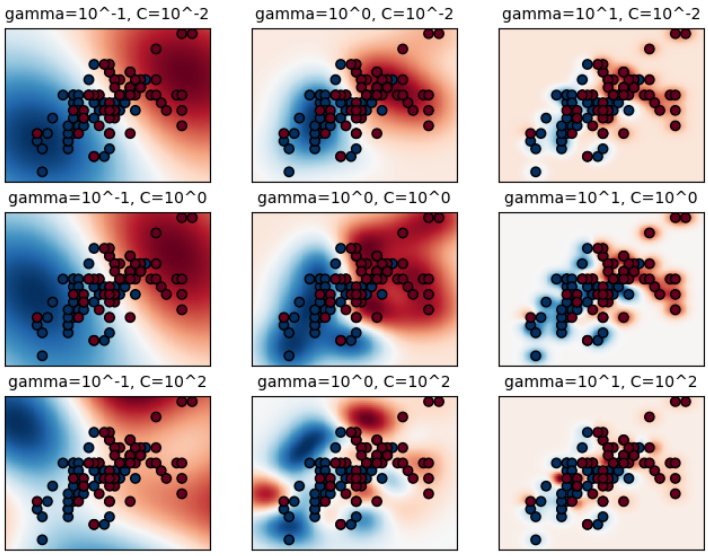


Abbildung 5.6: Einfluss von γ und C bei Support Vector Machines. Die Abbildung wurde von (Pedregosa et al., 2011a) übernommen.

In Abb. 5.6 ist der Einfluss von γ und von C am Beispiel einer Radial Basis Function (RBF) kernel SVM dargestellt. Es ist deutlich zu erkennen, dass ein hoher γ-Wert sich

negativ auf die Fähigkeit des Modells, zu generalisieren, auswirkt. Bei einem zu hoch gewählten γ verbessert auch der Regulierungsparameter C das Ergebnis nicht mehr (Pedregosa et al., 2011b).

Eigenes Vorgehen

Um die SVM zu implementieren wurde auf die scikit-learn Methode *SVC* (Support Vector Classification) zurückgegriffen. Diese wird speziell für Klassifikationsprobleme angewendet. SVC erhält ein Input-Array, welches die Features der Trainingssamples enthält, und ein Array welches die Klassenlabel enthält und wird daran mit der Funktion „fit“ angepasst. Danach können neue Werte der unbekannten Testsamples mit der Funktion „predict“ vorhergesagt werden (Pedregosa et al., 2011b).
Sowohl die Features der Trainings-, als auch die Features der Testdaten sind Z-skaliert (vgl. Abschnitt 4.3.2). Es wurden vier verschiedene Kernel-Funktionen ausprobiert: linear, polynomial, rbf und sigmoid. Insofern anwendbar, wurde ein Kernel-Koeffizient γ nach den Standardeinstellungen von scikit-learn von $1/(n_{features} * Var(X))$ verwendet. $n_{features}$ beschreibt dabei die Anzahl der eingegebenen Features, während $Var(X)$ die Varianz der Trainingsvektoren angibt. Da die Varianz der Trainingsdaten allerdings nach der Z-Skalierung 1 beträgt, kann diese ignoriert werden. Auch für den Regulierungsparameter C wurde die Standardeinstellung von 1,0 verwendet.
Es wurde auf die Standardeinstellungen zurück gegriffen, um zunächst einen allgemeinen Überblick der verschiedenen Lösungsansätze zu bekommen und die beste Strategie identifizieren zu können.

5.2.3 Naive Bayes

Der Naive Bayes-Klassifikator stellt eine probabilistische Lösung für Klassifikationsprobleme dar, die ursprünglich aus Arbeiten in der Mustererkennung hervorgegangen ist (Duda, 1973). Der Klassifikator beruht auf dem Satz von Bayes, der hier mit dem Klassenlabel K und dem Attributsvektor A_i bis A_n dargestellt ist:

$$P(K|A_1,...,A_n) = \frac{P(A_1,...,A_n|K) \cdot P(K)}{P(A_1,...,A_n)} \tag{5.5}$$

Um die Zugehörigkeit eines Datenpunktes zu einer Klasse ermitteln, wird für jede Klasse eine Zusammenfassung der Wahrscheinlichkeiten erstellt. Diese Zusammenfassung enthält zum einen die Wahrscheinlichkeit eines einzelnen Attributs A_i unter der Bedingung, dass es zu einer bestimmten Klasse K gehört ($P(A_i|K)$). Zum anderen enthält sie die a-priori Wahrscheinlichkeit der Klasse K ($P(K)$) sowie des jeweiligen Attributs A_i ($P(A_i)$). Diese Wahrscheinlichkeiten ergeben sich anhand der Trainingsdaten. Um herauszufinden, welcher Klasse ein Attribut aus dem Testdatensatz zugehört, wird der Satz von Bayes (Gl. 5.5) angewendet. Die a-posteriori Wahrscheinlichkeit $P(K|A_i)$ einer

bestimmten Klasse bei einem gegebenen Attribut errechnet sich aus den Wahrscheinlichkeiten, die anhand des Trainingsdatensatzes gewonnen wurden. Über die Formel 5.5 kann dann eine a-posteriori Gesamt-Wahrscheinlichkeit $P(K|A_1,...A_n)$ ermittelt werden. Der Klasse mit der höchsten a-posteriori Gesamt-Wahrscheinlichkeit wird letztendlich der neue Datenpunkt zugeordnet (Friedman, Geiger und Goldszmidt, 1997).
Die Berechnungen beruhen auf der Annahme, dass die einzelnen Attribute statistisch unabhängig voneinander sind. Das ist relativ unrealistisch, dennoch ist dieses Klassifikationsmodell praktisch anwendbar (Clark und Niblett, 1989). Das ist möglich, da es durch die „intrinsische Einfachheit“ des Modells zu einer geringen Varianz, also geringen Abweichungen, bei den vorhergesagten Wahrscheinlichkeiten kommt (Hand und Yu, 2001).

Eigenes Vorgehen:
In der vorliegenden Arbeit wurde die scikit-learn Methode *GaussianNB* verwendet. Diese implementiert den Gauß'schen Naive Bayes Algorithmus, bei dem davon ausgegangen wird, dass die Wahrscheinlichkeiten er einzelnen Attribute der Gauß-Verteilung unterliegen (Gl. 5.6). Die Parameter $\sigma_{K,i}$ und $\mu_{K,i}$ werden mittels Maximum Likelihood aus den Daten geschätzt (Pedregosa et al., 2011b). Die vorliegenden Features $\mathbf{x} = (x_1,...,x_n)^T$ entsprechen den zuvor beschriebenen Attributen A_i.

$$P(A_i|K) = \frac{1}{\sqrt{2\pi\sigma_{K,i}^2}} \exp\left(-\frac{(x_i-\mu_{K,i})^2}{2\sigma_{K,i}^2}\right) \tag{5.6}$$

5.3 Performanz und Validierung

5.3.1 Validierung der Modelle

Die einfachste Methode, ein Modell zu evaluieren, ist die **„Hold-Out-Methode“**. Dabei wird ein gelabelter Datensatz zufällig in zwei Teile geteilt, von dem ein Teil als Trainings- und der andere Teil als Testdatensatz verwendet wird. In der Regel ist der Trainingsdatensatz größer als der Testdatensatz. Der Trainingsdatensatz enthält sowohl alle Features als auch alle Label, während beim Testdatensatz die Label zwar bekannt sind, aber nur die Features genutzt werden. Das Modell wird dann auf den Trainingsdaten gelernt. Anschließend wird das gelernte Modell auf den Testdaten angewendet und versucht, die entsprechenden Klassenlabel vorherzusagen. Die Genauigkeit der Vorhersage kann dann mit den im nächsten Abschnitt 5.3.2 vorgestellten Metriken evaluiert werden. Wichtig ist dabei, dass das Modell nicht anhand der gleichen Daten evaluiert wird, auf denen es bereits trainiert wurde (diese Methode wird als **„Resubstitution Validation“** bezeichnet). In diesem Fall wäre die Gefahr groß, dass es zum sogenannten *Overfitting* kommt. Dabei würde ein Modell keine inhärenten Strukturen oder Muster in

den Daten lernen, sondern es würde lediglich die Trainingsdaten auswendig lernen. Es könnten dadurch keine generalisierenden Vorhersagen auf noch unbekannten Daten getroffen werden (Raschka, 2018).
Bei der Hold-Out-Methode werden Trainings- und Testdaten häufig zufällig aus dem Original-Datensatz gezogen, sodass bspw. $\frac{2}{3}$ der Daten zum Training und $\frac{1}{3}$ zum Testen genutzt werden. Durch diesen Prozess des *Random Subsamplings* wird allerdings die Annahme der statistischen Unabhängigkeit verletzt. Das bedeutet, wenn in einem Datensatz drei Klassen uniform verteilt sind (jede Klasse macht $\frac{1}{3}$ der Daten aus), kann es nach dem Random Subsampling dazu kommen, dass die Verteilungen der Klassen in Trainings- und Testdatensatz nicht mehr uniform verteilt sind. Die beiden Teilmengen repräsentieren also nicht mehr das Verhältnis der Klassen im Originaldatensatz. Dieses Problem wird noch drastischer bei Datensätzen mit großem Ungleichgewicht zwischen den Klassen. Hierbei kann es dazu kommen, dass die unterrepräsentierte Klasse überhaupt nicht im Testdatensatz auftaucht. Aus diesem Grund wird bei der Hold-Out-Methode häufig ein stratifizierender Ansatz verwendet, der dafür sorgt, dass die Klassenverhältnisse im Trainings- und im Testdatensatz korrekt repräsentiert werden (Kohavi, 1995; Raschka, 2018; Refaeilzadeh, Tang und Liu, 2009).

Die Hold-Out-Methode ist in der Anwendung nicht immer die geeignetste Methode, um ein Modell zu evaluieren. Oftmals sind die Datensätze von kleinem bis mittleren Umfang, und bei der Hold-Out-Methode wird ein Teil der Daten immer zurückgehalten, welcher nicht genutzt werden kann um das Modell zu trainieren. Es gibt Methoden, die die Daten effizienter nutzen, wie die **k-fold Cross-Validation**. Bei dieser Methode wird der Datensatz zufällig in k gleich große, sich nicht überschneidende Teilmengen geteilt. Das Modell wird k mal trainiert und getestet. In jedem Durchgang werden $k-1$ Teilmengen als Trainingsdaten verwendet, während der übriggebliebene Teil der Daten zum Testen genutzt wird. Am Ende wurde mit jeder Teilmenge ein Mal getestet und $k-1$ mal trainiert. Dabei wurden k Modelle an teilweise überlappenden Datensätzen trainiert, und an nicht überlappenden Validierungsdatensätzen getestet. Die Vorhersagegenauigkeit wird zum Schluss als arithmetisches Mittel aller k Metriken bestimmt (Raschka, 2018; Refaeilzadeh, Tang und Liu, 2009).
Um die Varianz des jeweiligen Performanzmaßes zu verringern, kann Cross-Validation mit unterschiedlichen k-fold Teilmengen wiederholt („r times repeated k-fold cross-validation“). Diese Wiederholungen reduzieren die Varianz allerdings nur geringfügig, wie Molinaro, Simon und Pfeiffer, 2005 zeigten.
Auch bei der k-fold Cross-Validation kann ein stratifizierender Ansatz verfolgt werden, der das mögliche Ungleichgewicht der Klassen berücksichtigt (Kohavi, 1995).

Eigenes Vorgehen:
In der vorliegenden Arbeit wurde eine 10-fold Cross-Validation angewendet, um die Modelle zu trainieren und zu evaluieren. Dazu wurde die *scikit-learn*-Methode *KFold* ver-

wendet. Auf eine Stratifikation wurde verzichtet, da die folds ausreichend viele Daten enthielten, sodass sowohl MLP-Daten von gesunden als auch von kranken Kühen abgedeckt waren.

5.3.2 Beurteilung der Performanz

Um die Qualität eines entwickelten Klassifikationsmodells beurteilen zu können, existieren eine Reihe an Evaluationsmaßen. Als Visualisierungshilfe wird häufig eine **Confusion Matrix** (dt.: Wahrheitsmatrix oder Konfusionsmatrix) verwendet, in der die vorhergesagten und die tatsächlichen Klassenlabel gegenübergestellt werden. In Abb. 5.7 ist exemplarisch eine Confusion Matrix dargetellt. Auf der Hauptdiagonalen ist die Anzahl der Elemente zu sehen, die der richtigen Klasse zugeordnet wurden. Diese Elemente werden *True Positives* (TP) genannt. Elemente außerhalb der Hauptdiagonalen wurden falsch zugeordnet und werden unterschieden in *False Positives* (FP) oder *False Negatives* (FN). Unter falsch positiven Ergebnissen versteht man Ergebnisse, die anzeigen, ein Zustand würde vorherrschen, obwohl er tatsächlich nicht vorherrscht. Ein falsch negatives Ergebnis beschreibt dementsprechend ein Ergebnis, dass die Abwesenheit eines Zustandes zeigt, obwohl dieser tatsächlich eingetreten ist (Rokach und Maimon, 2007) (Tharwat, 2020). Am Beispiel der Vorhersage einer Erkrankung würde ein falsch positives Ergebnis also bedeuten, dass eine Erkrankung fälschlicherweise diagnostiziert wurde, obwohl der Patient gesund ist. Bei einem falsch negativen Ergebnis bleibt die Erkrankung unerkannt (Tharwat, 2020).

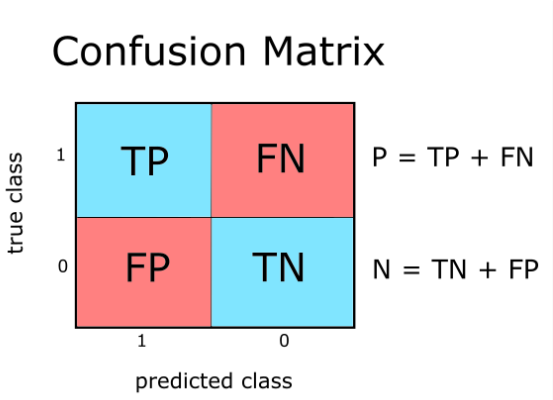


Abbildung 5.7: Allgemeine Darstellung einer Confusion Matrix. Verändert nach Tharwat, 2020.

Anhand der Confusion Matrix lässt sich die Berechnung diverser Evaluationsmaße gut nachvollziehen. Das wohl bekannteste Maß ist die **Accuracy**. Sie wird als Verhältnis der korrekt klassifizierten Datenpunkten zu der Gesamtzahl an Datenpunkten definiert (Gleichung 5.7). Komplementär dazu existiert die **Error-Rate**, bei der die inkorrekt klassifizierten Datenpunkte im Verhältnis zur Gesamtzahl an Datenpunkten betrachtet wird (Gleichung 5.8) (Tharwat, 2020).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{5.7}$$

$$Error-Rate = \frac{FP+FN}{TP+TN+FP+FN} = 1-Acc \tag{5.8}$$

Sowohl Accuracy als auch Error-Rate sind sensibel gegenüber ungleich verteilten Klassen. So kann ein Klassifikator zwar eine hohe Accuracy besitzen, aber in Wahrheit keinerlei Klassifizierung durchführen: es wird lediglich jeder Datenpunkt der überrepräsentierten Klasse zugeordnet (vgl. Abschnitt 4.6). Aus diesem Grund ist es unabdinglich, noch weitere Maße in die Evaluation eines Modells einzubeziehen (Tharwat, 2020).
Zwei Maße, die insbesondere bei medizinischen Diagnosen und Vorhersagen wichtig sind, sind **Sensitivity** (auch bezeichnet als True positve rate (TPR), Recall) und **Specifity** (auch True negative rate (TNR), inverse Recall). Sensitivity beschreibt das Verhältnis der korrekt klassifizierten positiven Datenpunkte zur Gesamtzahl an positiven Datenpunkten (Gleichung 5.9). Im Gegensatz dazu stellt Specifity das Verhältnis von korrekt klassifizierten negativen Datenpunkten zu allen negativen Datenpunkten dar (Gleichung 5.10). Beide Maße können die Qualität der Vorhersage bei unbalancierten Datensätzen verlässlich einschätzen (Tharwat, 2020).

$$Sensitivity = Recall = \frac{TP}{TP+FN} = \frac{TP}{P} \tag{5.9}$$

$$Specifity = \frac{TN}{TN+FP} = \frac{TN}{N} \tag{5.10}$$

Komplementär zu Sensitivity und Specifity existieren **False positive und false negative rates**. Die False positive rate (FPR, auch Fallout genannt) beschreibt das Verhältnis von inkorrekt klassifizierten negativen Datenpunkten zur Gesamtzahl der negativen Datenpunkte (Gl. 5.11). Die False negative rate (FNR) ist hingegen definiert durch das Verhältnis von positiven Datenpunkte, die inkorrekt klassifiziert wurden zur Gesamtzahl an positiven Datenpunkten (Gl.5.12). Auch diese Maße sind robust gegenüber Veränderungen in der Verteilung der Daten, weshalb sie ebenfalls bei unbalancierten Datensätzen Anwendung finden (Tharwat, 2020).

$$FPR = \frac{FP}{FP+TN} = \frac{FP}{P} = 1-Specifity \tag{5.11}$$

$$FNR = \frac{FN}{FN+TP} = \frac{FN}{P} = 1-Sensitivity \tag{5.12}$$

Darüber hinaus gibt es noch Maße, die speziell die Performanz der Vorhersage widerspiegeln. Die **Precision** (auch: positive prediction value (PPV)) repräsentiert den Anteil an positiven Datenpunkten, die korrekt klassifiziert wurden an der Gesamtzahl der posi-

tiv vorhergesagten Datenpunkte (Gl. 5.13). Gegenteilig dazu sagt die **Inverse Precision** (auch Negative predictive value (NPV) oder True negative accuracy (TNA)) aus, wie groß der Anteil der korrekt klassifizierten negativen Datenpunkte an der Gesamtzahl der als negativ vorhergesagten Datenpunkte ist (Gl. 5.14). Diese beiden Evaluationsmaße sind sensibel gegenüber ungleich stark repräsentierten Klassen (Tharwat, 2020).

$$Precision = PPV = \frac{TP}{FP + TP} \tag{5.13}$$

$$NPV = \frac{TN}{FN + TN} \tag{5.14}$$

Eigenes Vorgehen:
Da in der vorliegenden Arbeit das Verfahren der 10-fold Cross-Validation verwendet wurde, wurden für jeden der 10 Testdatensätze zunächst die jeweils absoluten TP-, TN-, FP- und FN-Anzahlen bestimmt. Dies geschah im Falle der Multi-Label Klassifikation mit der *scikit-learn*-Methode *multilabel_confusion_matrix*, im Falle der binären Klassifikation mit *confusion_matrix*. Die TP-, TN-, FP- und FP-Werte der 10 Durchgänge wurden im Anschluss aufsummiert um festzustellen, wie viele Instanzen des gesamten Datensatzes überhaupt korrekt positiv klassifiziert wurden etc. Anhand dieser Summen wurden dann die Evaluationsmaße Accuracy, Recall und Precision berechnet. Außerdem wurden die Werte innerhalb der Confusion Matrix nicht in absoluten Werten dargestellt, sondern jeweils als relativer Anteil aller positiven bzw. aller negativen Instanzen. Somit sind der Recall, die Specifity und die False Positive und False Negative rates ebenfalls direkt innerhalb der Matrix abzulesen.
Die Darstellung der Confusion Matrices wurde mit Hilfe von *seaborn*(V. 0.11.2) und *matplotlib* (V. 3.5.2) realisiert.

6 Ergebnisse

6.1 Überprüfen der ursprünglichen Problemstellung

Das endgültige Ziel der ursprünglichen Problemstellung besteht darin, anhand der letzten Milchleistungsprüfung einer Laktationsphase vorherzusagen, ob es in der darauf folgenden Laktationsphase in den ersten 30 bzw 100 Tagen zu einer Erkrankung kommt. Eine Visualisierung der ursprünglichen Problemstellung ist in Abbildung A.1 zu sehen.
Um zu Überprüfen, ob diese Problemstellung mit den gegebenen Daten zu lösen ist, werden zunächst die letzten MLP's jeder Laktation (Laktationsnr. X) mit R betrachtet und geprüft, ob es überhaupt eine darauf folgende Laktation (Laktationsnr. X+1) in den Daten gibt. Von 660 letzten MLPs war dies nur bei 256 der Fall. Im Anschluss daran wird geprüft, ob in der Laktationsphase X in den letzten 90 Tagen vor der letzten MLP irgendwelche Erkrankungen auftreten. Diese könnten unter Umständen die Inhaltsstoffe der Milch bereits soweit beeinflussen, dass eine Vorhersage darüber, ob eine Erkrankung in Zukunft eintritt oder nicht, unmöglich ist. Falls Erkrankungen in den 90 Tagen vor der letzten MLP auftreten, wird der Zustand „krank" und die Art der Erkrankung notiert. Daraufhin wird überprüft, ob Erkrankungen in den ersten 30 bzw 100 Tagen der Laktationsnr. X+1 auftraten. Auch hier wird der entsprechende Zustand „gesund" oder „krank" sowie die Art der Erkrankung notiert. Eine Visualisierung des Ansatzes ist in Abbildung 6.1 dargestellt. Die entstandene Übersicht ist in Tabelle 6.1 zu sehen. Dies wird zunächst mit dem gesamten Erkrankungsdatensatz durchgeführt.
Nachdem Informationen über relevante Erkrankungen vom LKS Lichtenwalde zu Verfügung gestellt wurden, wurde der Datensatz angepasst und die Problemstellung erneut überprüft. Welche Krankheiten innerhalb einer Laktation und über die Laktation hinaus einen Einfluss auf die MLP-Daten haben können, ist in Tabelle A4 dargestellt. Aus den jeweiligen Diagnosen ergibt sich, dass die folgenden Klassen für die Vorhersage relevant sind: SW1, SW3, BW1, BW2, BW3, EU1, TU1, TU2, SO1, ZH1, ZH3, ST. Dabei ist wichtig zu beachten, dass in den Krankheitsdaten nicht immer eine Klasse gegeben ist, weil die spezielle Diagnose keiner der vorhandenen Klassen zugeordnet werden konnte. Datenpunkte, bei denen das der Fall ist, werden in der Analyse zu den relevanten Erkrankungen nicht betrachtet. Außerdem werden die Krankheitsklassen PK1 und PK2 bei den Analysen nicht inkludiert, da deren Zusammenhang mit den MLP-Daten nicht sicher ist. Dadurch ergeben sich 2377 Erkrankungsdaten, mit denen gearbeitet werden kann.

Die genauen Zahlen über die Veränderung des Gesundheitszustands der Kühe von der einen zur nächsten Laktationsnummer, wenn alle Erkrankungsdaten mit einbezogen werden, ist in den oberen beiden Abbildungen von 6.2 zu sehen. Wenn nur die ersten 30 Tage der Laktation X+1 betrachtet werden, existieren deutlich mehr Kühe,

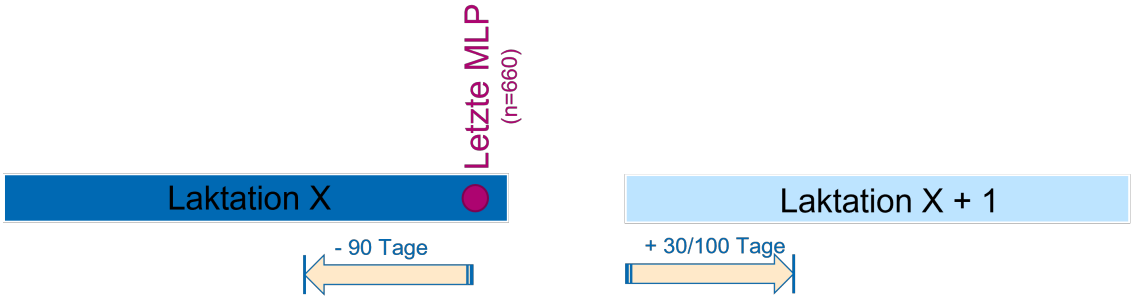


Abbildung 6.1: **Visualisierung des Ansatzes mit dem überprüft werden soll, ob die gegebene Problemstellung mit den vorhandenen Daten lösbar ist.** Es werden zunächst alle letzten MLPs der Laktationsnr. X einer Kuh ermittelt. Dabei handelt es sich um 660 Stück. Danach wird geprüft, ob bei dieser Kuh auf die betrachtete Lakt.-Nr. X eine weitere Lakt.-Nr. X+1 folgt. Dies ist in 256 Fällen der Fall.
Im Anschluss daran wird geprüft, ob in den 90 Tagen vor dem Datum der letzten MLP in Lakt.-Nr. X Erkrankungsdaten vorliegen und zu welcher Erkrankungsart diese gehören. Die jeweiligen Erkrankungsarten werden notiert. Zusätzlich wird geprüft, ob und welche Erkrankungen in den ersten 30 bzw. 100 Tagen der Lakt.-Nr. X+1 auftreten. Die jeweiligen Ergebnisse werden in einer Tabelle festgehalten, welche in Tabelle 6.1 visualisiert ist. Die Abbildung wurde in Microsoft 365 Powerpoint Version 2205 erstellt.

Infos über die MLP	Lakt_Nr_x	Status_x	Erkrankungen_x	Lakt_Nr_x+1	Status_x+1	Erkrankungen_x+1
...	4	krank	BW,EU,PA, SE,SW	5	krank	SW,BW
...	4	krank	BW	NA	NA	NA
...	3	krank	EU,PA,SE, SW	4	gesund	NA
...	4	gesund	NA	5	gesund	NA
...	...	...	...	...	...	...

Tabelle 6.1: **Ausschnitt der Übersicht darüber, ob die gegebene Problemstellung mit den vorhandenen Daten lösbar ist.** In den Informationen über die letzte MLP sind Daten über die Ohrnummer, das Kalbedatum, sowie das MLP-Datum enthalten. Lakt_Nr_x sagt aus, in welcher Laktationsnummer die betrachtete letzte MLP abgenommen wurde. Status_x beschreibt, ob in den 90 Tagen vor der letzten MLP die betrachtete Kuh in den Erkrankungsdaten auftaucht. Wenn ja, wird der Status „krank“ notiert und in der Spalte Erkrankungen_x die jeweils auftretenden Erkrankungsarten vermerkt. Sonst wird der Status „gesund“ notiert. Danach wird in Lakt_Nr_x+1 vermerkt, ob von dieser Kuh noch eine darauf folgende Laktation X+1 existiert. Wenn das der Fall ist, wird der Gesundheitszustand der Kuh in den ersten 30 bzw. 100 Tagen dieser Laktation notiert (Status_x+1), genauso wie die jeweils auftretenden Erkrankungsarten (Erkrankungen_x+1). Die Daten wurden in R Version 4.0.3 verarbeitet.

die während des gesamten Übergangs gesund bleiben (gg), als wenn die ersten 100 Tage betrachtet werden. Dies ist logisch, da bei der letzteren Variante ein mehr als drei-Mal so langer Zeitraum betrachtet wird, in dem dann folglich auch mehr Erkrankungen auftreten können. Es gibt insbesondere eine Steigerung in der Anzahl der Kühe, die zunächst gesund sind und dann in der darauf folgenden Laktation erkranken (+31 Fälle). Die Zahl der Kühe, die bereits in der vorausgehenden Laktation erkrankt waren und in der nachfolgenden Laktation erneut eine Erkrankung aufweisen, verändert sich nur geringfügig um 2 Exemplare.
Insbesondere die Werte von gg und gk wären für eine Vorhersage relevant. Von diesen existieren zusammengerechnet 173 Daten. Für das ausführliche Trainieren und Testen eines Machine Learning Modells mittels Cross-Validation ist das zu wenig.

Wenn nur die relevanten Erkrankungen betrachtet werden, werden deutlich mehr Kühe erwartet, die während des Übergangs von der einen zu anderen Laktation gesund bleiben (gg). Dies konnte auch so beobachtet werden (untere beiden Abbildungen von 6.2). Wenn der Gesundheitszustand der Kühe in den ersten 30 Tagen der Laktation betrachtet wird, zeigt sich ein starkes Ungleichgewicht der Klassen. Es existieren nun fast drei mal so viele Kühe, die während des gesamten Übergangs gesund bleiben (gg), wie Kühe, die zu Beginn der nachfolgenden Laktation erkranken (gk). Bei Betrachtung des Zustands in den ersten 100 Laktationstagen liegt ebenfalls ein Ungleichgewicht der Klassen vor, wenn auch ein nicht so gravierendes. Dies kann allerdings bei späteren Training und Testen eines Modell zu Problemen führen, wie in Abschnitt 4.6 bereits ausführlich dargelegt wird. Um das Klassenungleichgewicht auszubalancieren, müssten Sampling Methoden angewandt werden, welche den Umfang der verfügbaren Daten wiederum verringern könnten. Außerdem würde es sich schwierig gestalten, bei diesem geringen Umfang an Daten ein Modell zu trainieren, welches eine konkrete Erkrankungsart vorhersagen kann. Bei sieben relevanten Erkrankungsarten würden zu wenig Trainingsdaten für jede Erkrankungsart vorliegen. Lediglich eine allgemeine Vorhersage darüber, ob eine Kuh zu Beginn der nächsten Laktation erkrankt oder gesund bleibt, wäre möglich. Aus diesem Grund wird die Problemstellung angepasst.

6.2 Anpassung der Problemstellung

Bei der angepassten Problemstellung wird lediglich betrachtet, welche Erkrankungen im Zeitraum zwischen zwei aufeinander folgenden Milchleistungsprüfungen auftraten. Damit soll geprüft werden, ob anhand der MLP-Daten eine Vorhersage über die in naher Zukunft auftretenden Krankheiten gemacht werden kann. Mittels Binarisierung wird ein Datensatz erschafft, der zu jeder MLP die Erkrankungsarten zuordnet, die im Zeitraum bis zur nächsten MLP im Erkrankungsdatensatz zu finden sind. Der Aufbau des Datensatzes bei Betrachtung der relevanten Erkrankungen kann in Tabelle 6.2 nachvollzogen werden. Das ist auch der Datensatz, der für alle nachfolgenden Analysen verwendet

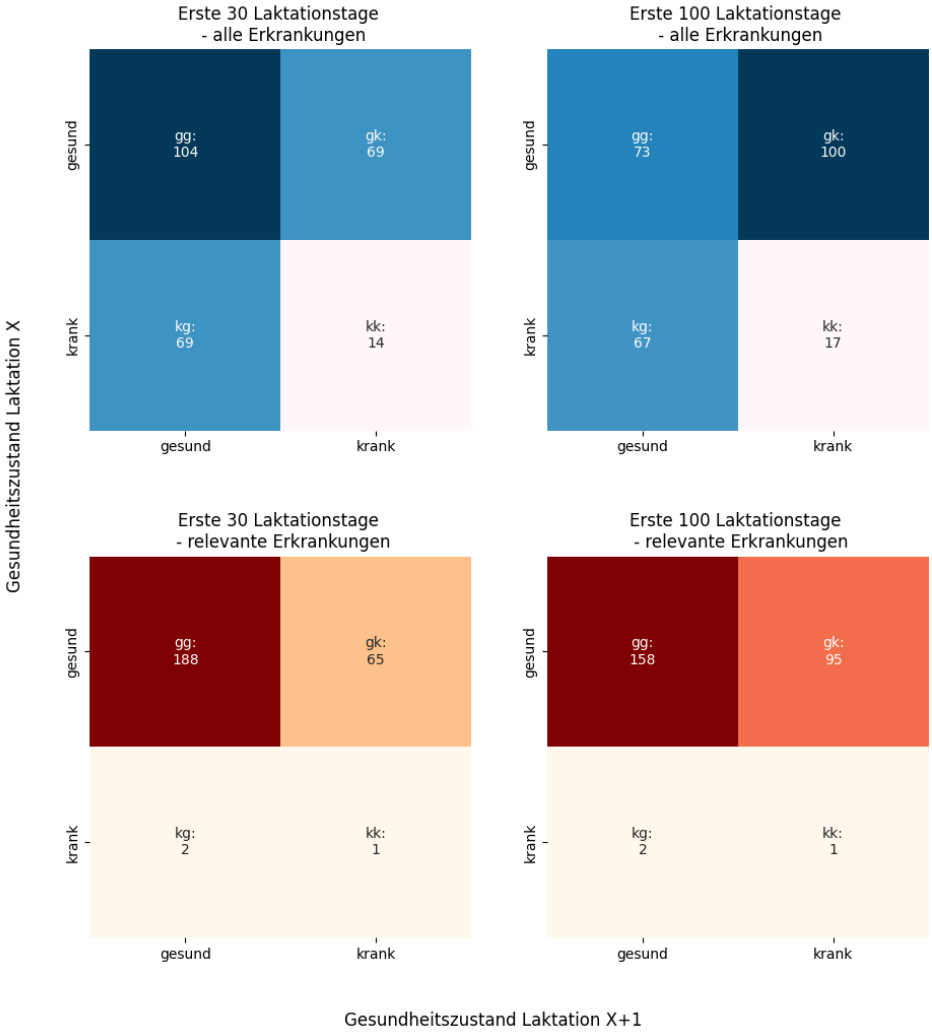


Abbildung 6.2: Veränderung des Gesundheitszustands der Kühe innerhalb von zwei aufeinander folgenden Laktationen.

wird.

Neben den Daten der Milchleistung finden sich die Erkrankungsdaten. Sie sind in 10 Spalten aufgeteilt bzw. in 7 Spalten, wenn nur die relevanten Erkrankungen betrachtet werden. Diese Spalten sind gefüllt mit 0, wenn im Zeitraum bis zu nächsten MLP kein Eintrag dieser Kuh in den Erkrankungsdaten zu finden ist, und mit 1, wenn doch. Wichtig hierbei ist, dass die Zeiträume zwischen zwei MLPs sehr variabel sind, wie Abb. A.2 verdeutlicht. Der Abstand reicht von 20 Tagen bis hin zu fast 65 Tagen. In der Abbildung wird unterschieden, ob im Zeitraum zwischen den beiden MLPs mindestens eine Erkrankung auftritt („erkrankt“) oder nicht („gesund“). Durch diese Unterscheidung soll überprüft werden, ob sich die unterschiedlich großen Abstände zwischen zwei MLPs dadurch erklären lassen, dass beim Auftreten einer Erkrankung möglicherweise die Milchleistungsprüfung für einen Monat ausgesetzt wird. Da aber in beiden Fällen Abstände von bis zu 65 Tagen auftreten, kann diese Annahme verworfen werden.

Abgesehen davon muss beim finalen Datensatz beachtet werden, dass alle Erkran-

kungsdaten der noch verbleibenden Laktation hinzugefügt werden, wenn es sich um die letzte MLP einer Laktation handelt. Unter Umständen können dadurch Erkrankungen hinzugefügt werden, die erst sehr viel später nach der letzten MLP auftraten.
Dadurch ergeben sich letztendlich 509 MLP's, denen mindestens eine Erkrankungsart zugeordnet wird. Bei 3563 MLPs fand keine Erkrankung im Zeitraum bis zur nächsten MLP statt. Warum nur nach einem Bruchteil der MLPs eine Erkrankung auftrat, lässt sich anhand Abbildung A.3 eindeutig erkennen: Ein Großteil der Erkrankungen findet nicht zwischen den Milchleistungsprüfungen statt, sondern bereits davor. Dies scheint unabhängig von der Art der Erkrankung zu sein. Außerdem fällt auf, dass die Erkrankungsart „SO“ kein einziges Mal während der Milchleistungsprüfungen auftritt. Aus diesem Grund wird sie bei den nachfolgenden Vorhersagen ignoriert.

Ohrnummer	**Laktationstag**	...	**Lakt_Nr**	**BW**	**SW**	**ST**	**SO**	**TU**	**EU**	**ZH**
1404584765	133	...	4	0	0	0	0	0	0	0
1404584765	168	...	4	0	0	0	0	0	0	0
1404584765	196	...	4	0	0	0	0	0	0	0
1404584765	231	...	4	1	0	0	0	0	0	0
1404584765	287	...	4	0	0	0	0	0	0	0
1404584765	17	...	5	1	1	0	0	0	0	0
1404584776	40	...	4	0	0	0	0	0	0	0

Tabelle 6.2: Aufbau des finalen Datensatzes bei Betrachtung der relevanten Erkrankungen.

6.3 Vorhersagen

Im Folgenden wird versucht, das Auftreten einer Erkrankung im Zeitraum bis zur nächsten MLP vorherzusagen. Dabei werden sowohl Multi-Label als auch binäre Klassifikationen durchgeführt. Für jedes Klassifikationsverfahren wird zunächst der unbalancierte Datensatz betrachtet, welcher 4070 Instanzen beinhaltet. Daran anschließend wird die Vorhersage erneut anhand des balancierten Datensatzes durchgeführt. Hierbei unterscheidet sich die Größe des Datensatzes je nach Anzahl der Erkrankungsdaten. Eine genaue Erklärung, wie die Daten balanciert werden, findet sich in Abschnitt 4.6. Bei der Multi-Label Klassifikation wird sowohl für den unbalancierten als auch den balancierten Datensatz wird eine Vorhersage alle relevanten Erkrankungsarten getroffen. „Sonstige Erkrankungen“ (SO) werden dabei vernachlässigt, da von diesen keine Daten in den betrachteten Zeiträumen vorhanden sind. Außerdem wird versucht, eine Vorhersage über den allgemeinen Gesundheitszustand (bleibt gesund oder erkrankt) zu machen. Bei der binären Klassifikation werden neben dem allgemeinen Gesundheitszustand nur die Erkrankungsarten EU, BW und ST betrachtet, da von den restlichen Arten nicht genügend Daten vorlagen.
Bei jedem Modell werden die folgenden Features genutzt: 'Laktationstag', 'Milchmenge', 'Fett', 'Protein', 'FEQ', 'Eiweiss_Energieverhaeltnis', 'Emin', 'Emax', 'Fmin', 'Fmax', 'Harnstoff', 'Laktose', 'pH', 'Zellen', 'C14_0', 'C16_0', 'C18_0', 'C18_1', 'Saturated_FA',

'Total_Unsaturated_FA', 'Mono_Unsaturated_FA', 'Poly_Unsaturated_FA', 'SCFA', 'MCFA', 'LCFA', 'Trans_FA', 'De_novo_FA', 'Mixed_FA', 'Preformed_FA', 'De_novo_FA_FA', 'Mixed_FA_FA', 'Preformed_FA_FA', 'De_novo_FA_g_Tag', 'Mixed_FA_g_Tag', 'Preformed_FA_g_Tag' und 'Lakt_Nr'. Sie werden wie in Abschnitt 4.3.2 erläutert, Z-skaliert. Die Klassenlabel bilden die jeweils betrachteten Erkrankungsarten. Die Confusion Matrices zu allen Vorhersagen sind im Anhang zu finden. Im Folgenden werden nur die Ergebnisse ausgewählter Modelle dargestellt. Eine Erklärung zu den Confusion Matrices ist in Abschnitt 5.3.2 nachzulesen.

6.3.1 Multi-Label Klassifikation

Um die Multi-Label Klassifikation durchzuführen, wird ein Multi-Layer Perzeptron verwendet, wie es in Abschnitt 5.2.1 beschrieben wird.
Sowohl vor als auch nach dem Balancieren des Datensatzes werden fast alle Datenpunkte als 0, also gesund bleibend vorhergesagt (Abb. 6.3 und A.4). Die einzige Veränderung in der Vorhersage nachdem der Datensatz balanciert wurde, lässt sich bei der Erkrankungsart BW beobachten (Abb. 6.3). Dort werden nach dem Undersampling deutlich mehr Datenpunkte als True Positive klassifiziert, gleichzeitig werden aber auch weniger True Negatives vorhergesagt. Insgesamt ist die Vorhersage von Erkrankungen des Bewegungsapparats immer noch sehr ungenau, da nur 29,49 % der Erkrankungen zuverlässig vorhergesagt werden können. Bei den restlichen Erkrankungsarten zeichnet sich ein noch schlechteres Bild, dort werden oft fast alle Datenpunkte als gesund vorhergesagt, auch nach dem Undersampling.

6.3.2 Binäre Klassifikation

Bei der binären Klassifikation wird der Datensatz so verändert, dass nur noch eine Erkrankungsart betrachtet wird. Für jede Erkrankungsart wird also ein eigenes Modell trainiert, wodurch andere Klassifikatoren nutzbar sind wie Naive Bayes oder SVM. Um den Datensatz für die binäre Klassifikation zu kreieren werden nur die MLPs genutzt, denen entweder die betrachtete Erkrankungsart zugeordnet wird oder denen gar keine Erkrankungsart zugeordnet wird. Wenn zusätzlich zu der betrachteten Art noch eine andere Art auftritt, wird die MLP dennoch genutzt. Wenn allerdings statt der betrachteten Art eine oder mehrere andere Erkrankungsarten auftreten, werden jene MLP-Daten entfernt. Dabei wird sich auf die drei Erkrankungsarten „BW“, „EU“ und „ST“ beschränkt. Außerdem wird versucht, den allgemeinen Gesundheitszustand der Kuh vorherzusagen („krank“). Dabei wird betrachtet, ob im Zeitraum bis zur nächsten MLP irgendeine der relevanten Erkrankungen auftritt oder nicht.

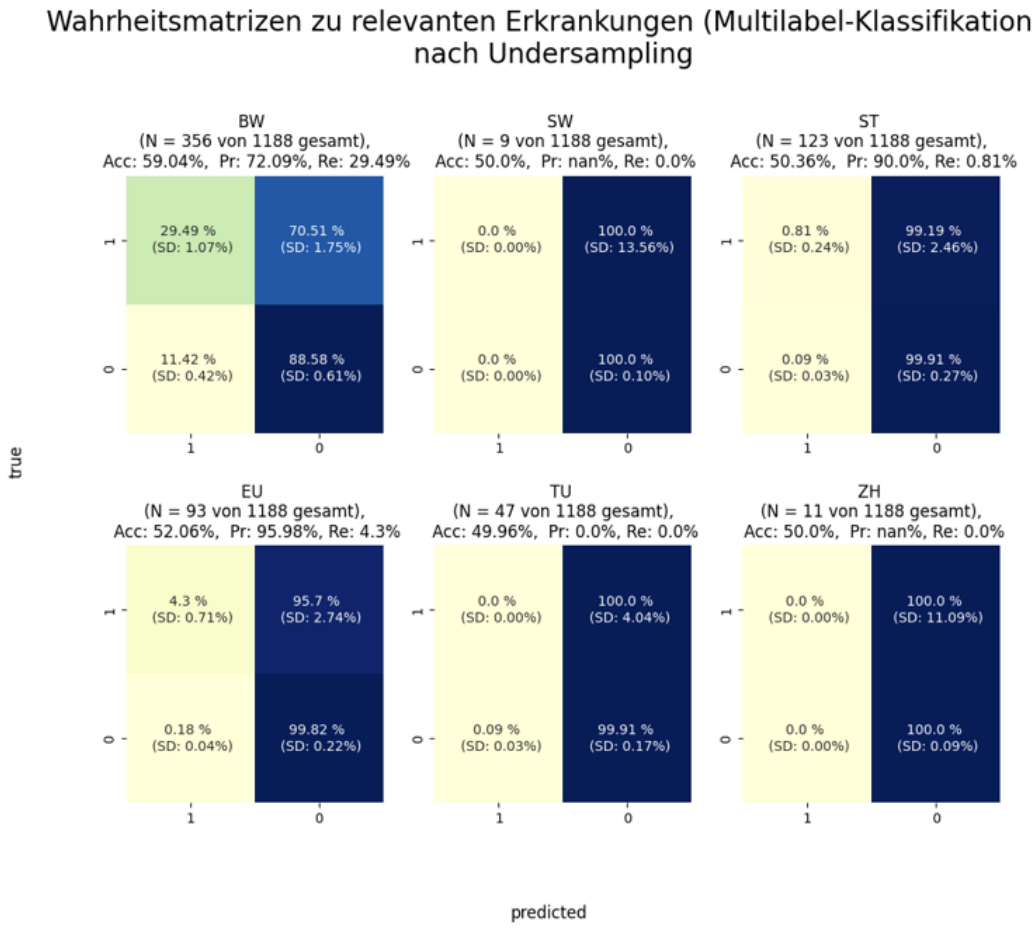


Abbildung 6.3: Ergebnisse der Multi-Label-Klassifikation des Multi-Layer Perzeptrons auf balancierten Daten.

Multi-Layer Perzeptron

Bei der binären Klassifikation wird zunächst ebenfalls ein Multi-Layer Perzeptron verwendet. Vor der Balancierung des Datensatzes konnte das Multi-Layer Perzeptron keine Klassifzierung durchführen, alle Datenpunkte werden als gesund vorhergesagt (Abb. A.6a). Nach dem Undersampling verbesserte sich die Performance des Modells stark (Abb. 6.4). Insbesondere bei der Vorhersage des allgemeinen Gesundheitszustands und bei der Vorhersage von Erkrankungen des Bewegungsapparats (BW) werden zwei Drittel bzw drei Viertel der tatsächlich in naher Zukunft erkrankenden Kühe als solche vorhergesagt.

Naive Bayes

Neben dem Multi-Layer Perzeptron wird außerdem der Naive Bayes Klassifikator genutzt. Dieser erreicht auch bei unbalancierten Daten eine bessere Performance als andere Modelle vor dem Undersampling (vgl. Abb. 6.5a). Bald auftretende Störungen der Trächtigkeit (ST) können in 71,54 % der Fälle korrekt vorhergesagt werden, wobei anzu-

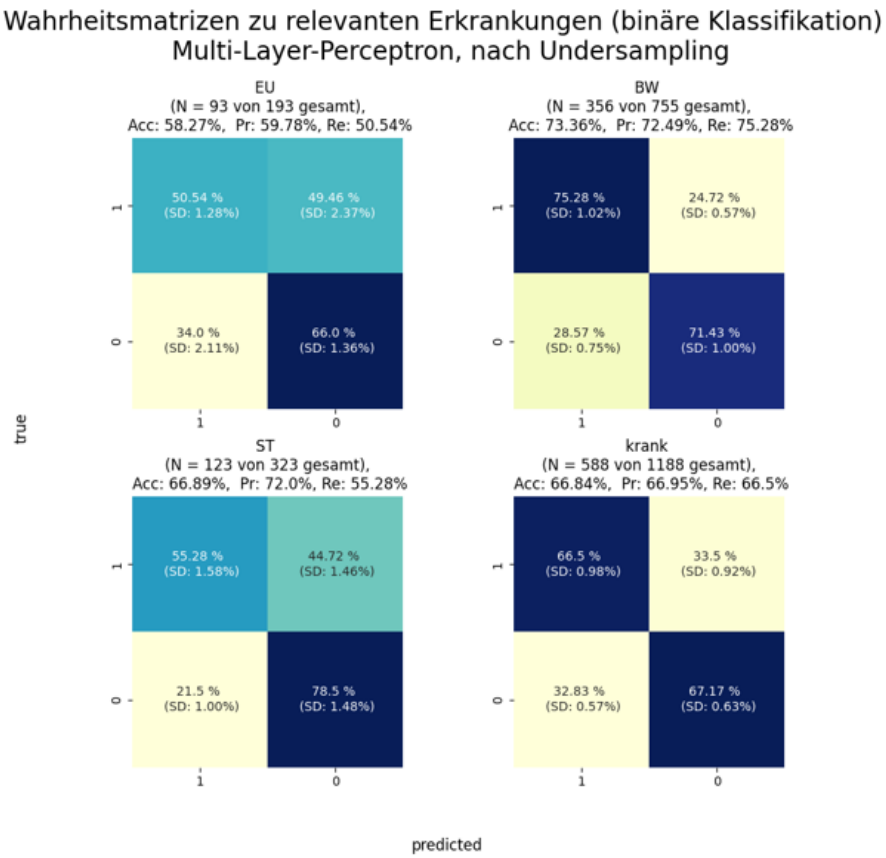
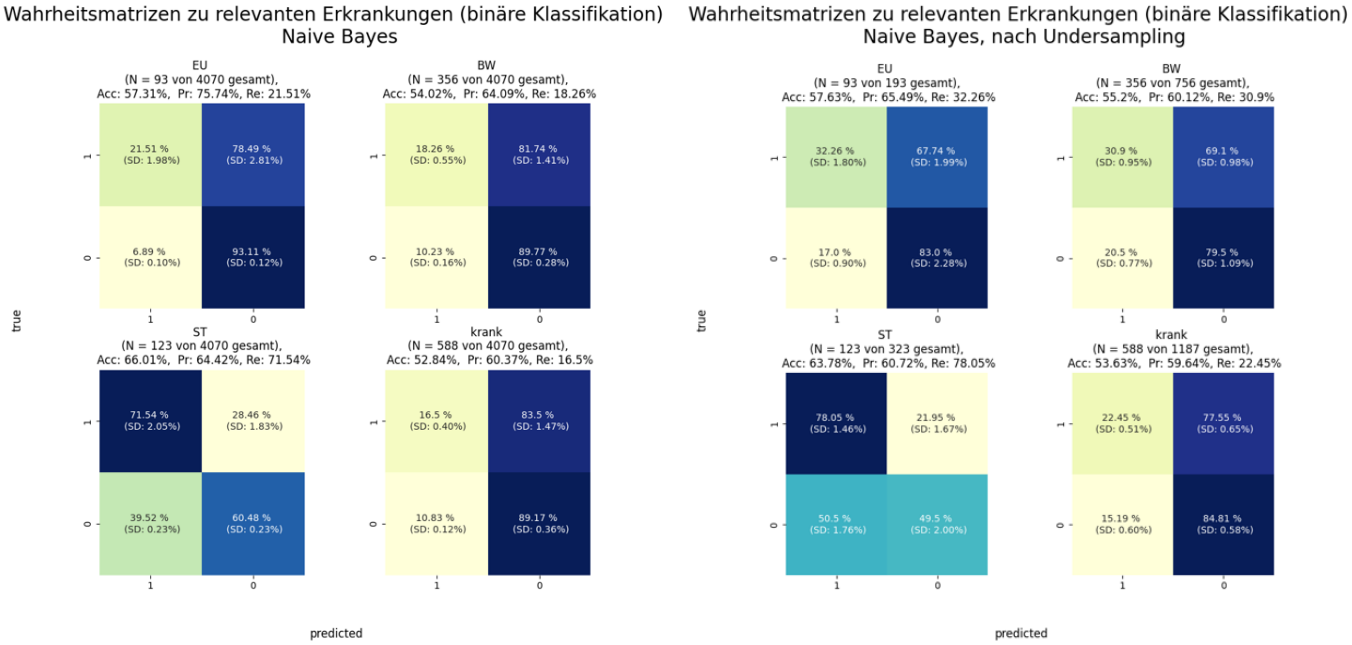


Abbildung 6.4: Ergebnisse der binären Klassifikation des Multi-Layer Perzeptrons auf balancierten Daten.

merken ist, dass auch bei 39,53 % der gesund bleibenden Kühe fälschlicherweise eine Erkrankung vorhergesagt wird. Erstaunlicherweise verschlechtert sich die Vorhersagegenauigkeit des Naive Bayes Klassifikators nach dem Undersampling bei den Störungen der Trächtigkeit noch (vgl. Abb. 6.5b). Es wird in diesem Fall sogar der Hälfte aller Kühe eine falsch-positive Vorhersage gegeben, während die Zahl der falsch-negativen Vorhersagen in etwa gleich bleibt. Bei den anderen Erkrankungsarten und dem allgemeinen Gesundheitszustand hat sich die Vorhersagegenauigkeit etwas verbessert. Dennoch sind die Vorhersagen nicht so zuverlässig wie die des Multi-Layer Perzeptrons nach dem Undersampling.

Support Vector Machine

Als dritter Klassifikator wird eine Support Vector Machine verwendet, bei der vier verschiedene Kernels getestet werden. Anhand des unbalancierten Datensatzes werden bei allen Kernels beinahe alle Datenpunkte als gesund klassifiziert (Abb. A.8). Nach dem Undersampling verbessert sich die Vorhersage etwas (Abb. A.9) . Insbesondere die Vorhersagen der SVM mit linearem Kernel (Abb. 6.6a) über den allgemeinen Ge-



(a) Vorhersagergebnisse beim unbalancierten Datensatz.

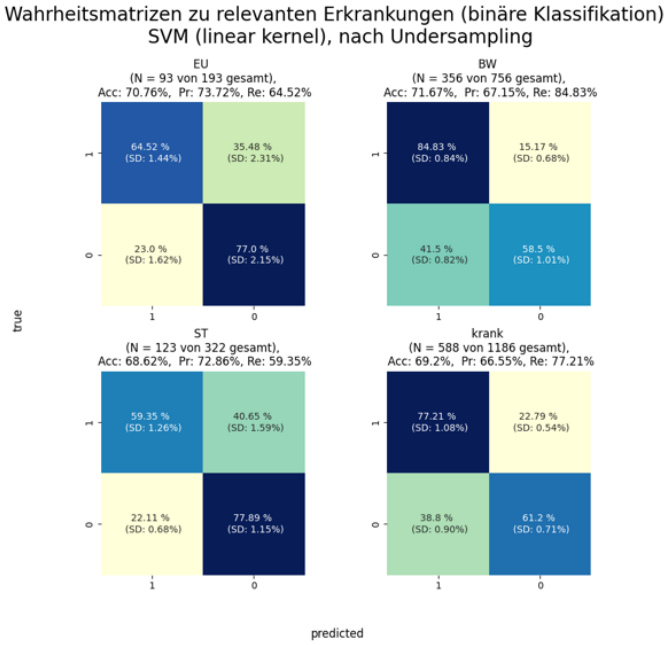
(b) Vorhersagergebnisse beim balancierten Datensatz.

Abbildung 6.5: Ergebnisse der binären Klassifikation durch Naive Bayes auf unbalancierten und balancierten Daten.

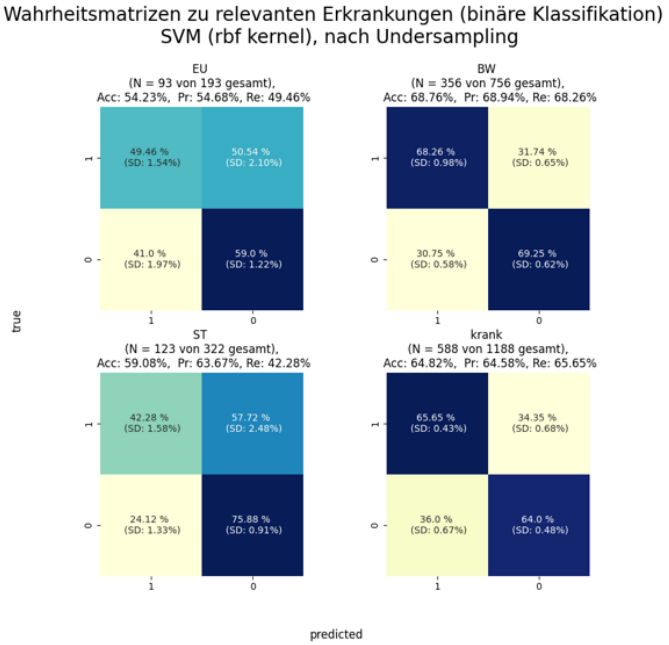
sundheitszustand und über Erkrankungen des Bewegungsapparats sind genauer als erwartet. Eine etwas schlechtere Vorhersage bietet die SVM mit RBF Kernel (Abb. 6.6b) Andere Modelle scheinen hingegen kaum eine Klassifikation durchführen zu können, die besser ist als zufälliges Raten, wie bspw. die SVM mit sigmoid Kernel (Abb. A.9d).

6.3.3 Einbeziehen der Historie

Im Folgenden soll die Veränderung der MLP-Parameter im Verlauf der Laktation mit einbezogen werden, um die Vorhersagen einer Erkrankung genauer zu machen. Es soll damit nicht nur eine „Momentaufnahme“ der MLP-Werte genutzt werden, sondern die vorausgehende Veränderung und die Abweichung vom durchschnittlichen Verlauf der MLP-Werte berücksichtigt werden. Das beispielhafte Vorgehen ist in Abbildung 6.7 dargestellt.
Zunächst werden zu jedem Laktationstag die Mittelwerte all derer MLP-Daten ermittelt, denen keine Erkrankungsart zugeordnet werden kann. Im Optimalfall kann so für jeden Parameter zu jedem Laktationstag ein Mittelwert gebildet werden, damit sich eine Art durchschnittliche Verlaufskurve jedes einzelnen MLP-Parameters ergibt.
Im Anschluss daran wird die individuelle Kuh betrachtet und die Differenz aller MLP-Parameter zwischen der MLP i und der vorhergehenden MLP i-1 ermittelt. Am Beispiel des MLP-Parameters Laktose sieht die Berechnung wie folgt aus: $\Delta Lakt = Lakt_{i-1} - Lakt_i$. Zwischen dem durchschnittlichen MLP-Parameter des Laktationstags, an dem



(a) Confusion Matrices der SVM mit linearem Kernel.



(b) Confusion Matrices der SVM mit rbf Kernel.

Abbildung 6.6: Ergebnisse der binären Klassifikation durch SVMs mit linearem und rbf Kernel. Die Modelle wurden auf dem balancierten Datensatz angewendet.

MLP i aufgenommen wurde, und dem durchschnittlichen MLP-Parameter des Laktationstags, an dem die vorhergehende MLP i-1 bestimmt wurde, wird ebenfalls eine Differenz gebildet ($\Delta\hat{Lakt} = \hat{Lakt}_{i-1} - \hat{Lakt}_i$). Die endgültige Abweichung des MLP-Parameters ergibt sich, indem die Differenz der beiden soeben bestimmten Differenzen berechnet wird ($\Delta Lakt - \Delta\hat{Lakt}$). Es wird also die Abweichung der Veränderung von der Differenz der Durchschnittswerte bestimmt.

Allerdings konnte nicht zu jedem Laktationstag ein Mittelwert berechnet werden, insbesondere sehr früh und sehr spät in der Laktation sind an einigen Tagen oft keine MLP-Daten aufgenommen worden. Dies ist an 18 Tagen der Fall: 1, 2, 3, 4, 424, 440, 448, 465, 476, 501, 536, 565, 567, 581, 597, 625, 710 und 712. Da nur von den „gesunden“ MLPs Mittelwerte bestimmt wurden, kann es vorkommen, dass von einigen MLPs, denen eine Erkrankung zugeordnet wurde, keine Abweichung bestimmt werden kann, da diese an einem dieser „fehlenden“ Laktationstage aufgenommen wurde. So gehen unter Umständen wichtige Daten verloren.
Außerdem gehen 660 MLPs verloren, weil es sich bei ihnen um die erste MLP der Laktation handelt. Bei ihnen kann keine Differenz zur vorherigen MLP der gleichen Laktation gebildet werden. Aus diesem Grund fällt die Zahl der beobachteten Erkrankungen bei den historischen Daten kleiner aus als bei den bloßen MLP-Daten.
Bei den veränderten Daten werden dieselben Klassifikationsmodelle genutzt, wie bereits bei den unveränderten MLP-Daten beschrieben.

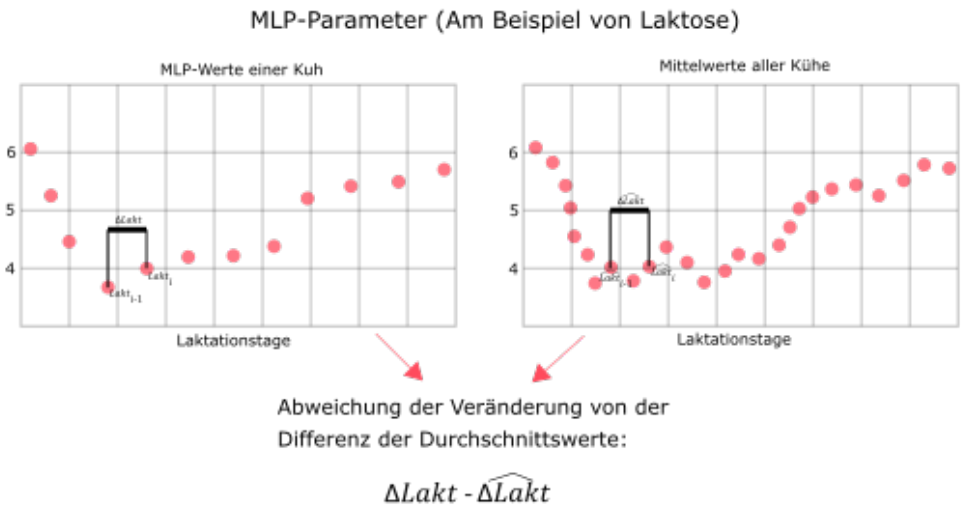


Abbildung 6.7: Visualisierung davon, wie Informationen über die vorangegangenen MLPs einbezogen werden, um einen veränderten Datensatz zu erzeugen.

Bei der Multi-Label Klassifikation ist die Performanz des Modells genauso schlecht wie die des Modells mit den unveränderten MLP-Daten. Sowohl vor als auch nach dem Undersampling werden alle Daten als „gesund“ vorhergesagt (Abb. A.10 und A.11).

Bei der binären Vorhersage mittels Multi-Layer Perzeptron ist die Vorhersage vor dem Undersampling ebenfalls nichtssagend. Nach dem Undersampling verbessert sich die Vorhersage gegenüber den unveränderten MLP-Daten nicht. Es werden bei allen Erkrankungsarten und beim allgemeinen Gesundheitszustand weniger True Positive Ergebnisse erreicht, als bei den unveränderten MLP-Daten. Dafür werden mehr True Negative Ergebnisse vorhergesagt (Abb. 6.8).

Der Naive Bayes Klassifikator gibt vor dem Undersampling eine ähnliche Vorhersage wie bei den unveränderten MLP-Daten. Die genaue Vorhersage bei Störungen der Trächtigkeit (ST) ist allerdings vollkommen verloren gegangen (Abb. 6.9a). Nach dem Undersampling verbessert sich die Performanz des Klassifikators etwas, allerdings kommt es trotzdem noch zu 40% oder mehr falsch-negativen Ergebnissen (Abb. 6.9b).

Ohne Balancing der Daten versagen auch bei den historischen Daten die Vorhersagen aller genutzten SVMs. (Abb. A.14). Nach dem Balancing fällt es den SVMs mit polynomialem und sigmoid Kernel immer noch am schwersten, auftretende Erkrankungen bis zur nächsten MLP vorherzusagen. Die SVMs mit linearem und rbf Kernel, die bei den unveränderten MLP-Daten bei einigen Erkrankungsarten bzw. dem allgemeinen Ge-

sundheitszustand eine zuverlässigere Vorhersage gaben, ließen bei den historischen MLP-Daten in ihrer Performanz nach. Fast immer konnte ein Großteil der positiven Ereignisse nicht vorhergesagt werden (Abb. A.15).

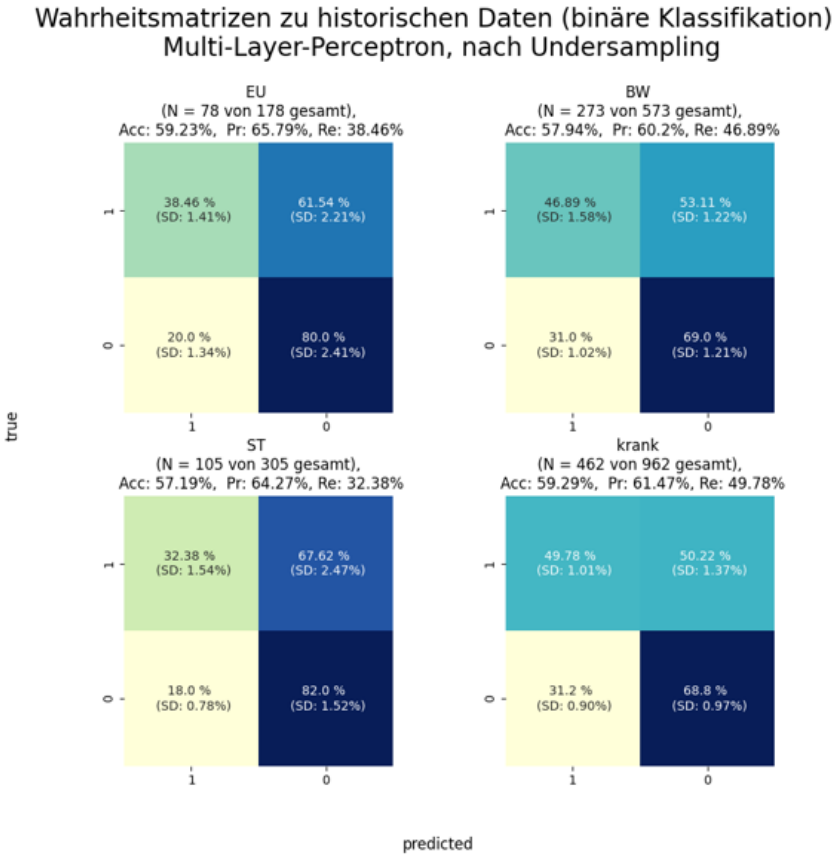
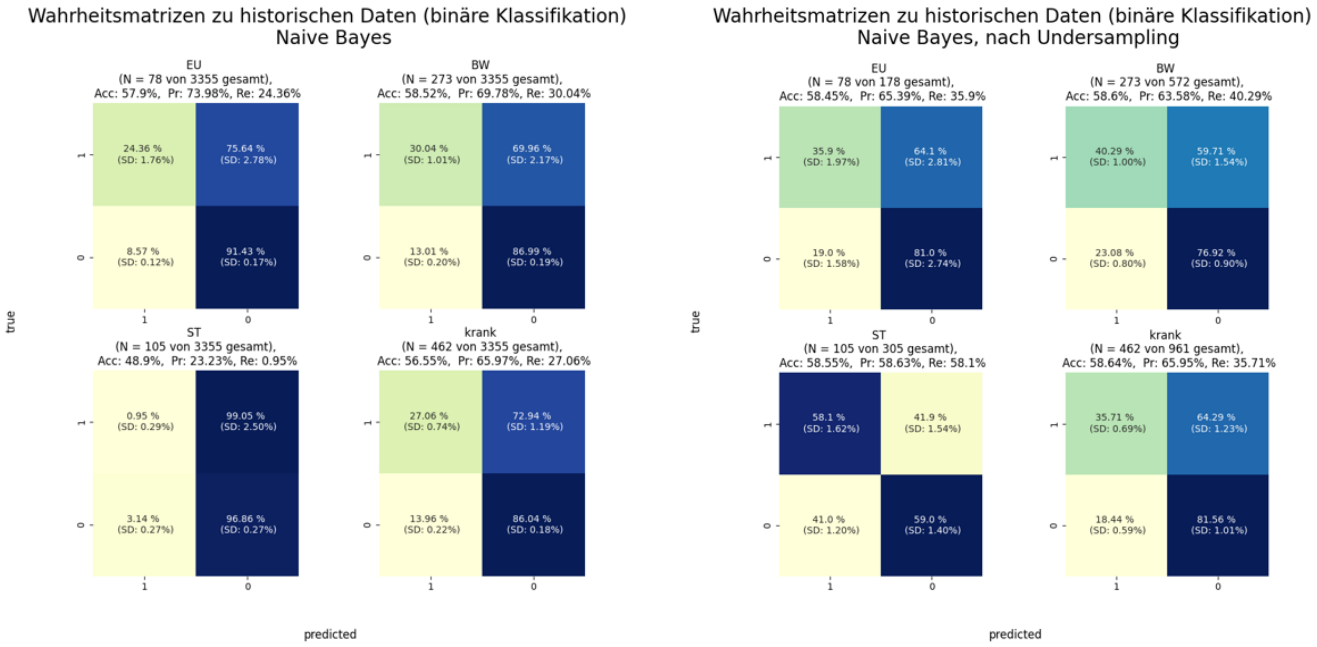


Abbildung 6.8: Ergebnisse der binären Klassifikation des Multi-Layer Perzeptrons auf balancierten historischen MLP-Daten.



(a) Vorhersagergebnisse beim unbalancierten Datensatz.

(b) Vorhersagergebnisse beim balancierten Datensatz.

Abbildung 6.9: Ergebnisse der binären Klassifikation durch Naive Bayes auf unbalancierten und balancierten historischen MLP-Daten.

7 Auswertung und Diskussion

7.1 Bewertung der Modelle

Welches Modell am geeignetsten für die Anwendung ist, steht und fällt mit den Zielen und Prioritäten. Im Fall der vorliegenden Arbeit sollen möglichst viele der Erkrankungen vorhergesagt werden, die im Zeitraum bis zur nächsten MLP auftreten könnten. Das bedeutet, die Zahl der True-Positive Ergebnisse sollte möglichst hoch ausfallen. Gleichzeitig sollte die Zahl der Falsch-Negativen Ergebnisse, bei denen eine bevorstehende Erkrankung nicht erkannt wird, möglichst minimiert werden. Genauso sollten aber auch möglichst wenige Vorhersagen als False Positive eingeordnet werden. Sonst würden in diesem Fall Kühe, die eigentlich gesund bleiben, aufgrund einer fälschlicherweise vermuteten Erkrankung näher beobachtet werden. Das kann zu einem erhöhtem Aufwand und erhöhten Kosten führen. Letztendlich ist es eine Frage der Prioritäten, ob eine hohe Zahl an Falsch-Positiven Ergebnissen in Kauf genommen wird, wenn dafür mehr Erkrankungen frühzeitig erkannt werden, oder ob ein gewisser Teil der Erkrankungen unerkannt bleibt, wenn dafür Kosten und Aufwand eingespart werden.
Ein geeignetes Modell, um das Risiko aller Erkrankungsarten auf einmal vorherzusagen, konnte ein dieser Arbeit nicht gefunden werden. Dafür wäre die Multi-Label Klassifikation in Frage gekommen, allerdings weist das Modell dazu sogar nach dem Balancieren des Datensatzes nur eine geringe Vorhersagegenauigkeit auf. Ein Großteil oder fast alle Datenpunkte werden als gesund vorhergesagt. Das liegt zum Teil daran, dass die verschiedenen Erkrankungsarten unterschiedlich häufig auftreten, was ein sinnvolles Undersampling erschwerte.
Die Modelle, die immer nur eine Erkrankung binär klassifizieren, weisen hingegen eine bessere Performanz auf. In jedem Fall, außer beim Naive Bayes Klassifikator, ist Undersampling nötig, um brauchbare Ergebnisse zu erhalten. Außerdem sollte auf die reinen Z-skalierten MLP-Daten zurückgegriffen werden, da das Einbeziehen von historischen Daten die Vorhersagegenauigkeit eher noch verschlechtert.

Nach dem Undersampling werden beim Versuch der Vorhersage des allgemeinen Gesundheitszustandes bei einigen binären Klassifikatoren gute bis sehr gute Ergebnisse erzielt. Insbesondere das Multi-Layer-Perzeptron oder eine SVM mit linearem oder rbf Kernel könnten hier am geeignetsten sein. Bei allen drei Modellen werden mehr als zwei Drittel der in naher Zukunft erkrankenden Kühe korrekt positiv vorhergesagt. Genauso werden nahezu zwei Drittel oder mehr der gesund bleibenden Kühe als solche erkannt. Aber auch die Falsch-Negativ-Rate muss bei der Vorhersage von Erkrankungen berücksichtigt werden, d.h. wie viele der bald erkrankenden Kühe nicht als solche erkannt werden. Beim Multi-Layer Perzeptron und bei der SVM mit rbf Kernel wird etwa ein Drittel dieser Fälle nicht erkannt, während es bei der SVM mit linearem Kernel le-

diglich knapp 23% sind. Gleichzeitig werden bei allen drei Modellen nahezu oder mehr als ein Drittel der gesunden Kühe als fälschlicherweise erkrankend klassifiziert. Da sich diese Falsch-Positiv-Rate nur geringfügig zwischen den betrachteten Modellen unterscheidet, ist die SVM mit linearem Kernel der beste Kandidat, um Vorhersagen über den allgemeinen Gesundheitszustand zu treffen.
Erkrankungen des Bewegungsapparats (BW) können von diesen drei Modellen ebenfalls vorhergesagt werden. Während die SVM mit linearem Kernel mehr als 84% True-Positive Ergebnisse aufweist und nur 15% falsch-negative Vorhersagen trifft, ist die Genauigkeit bei den negativen Instanzen eher schwach: 41% der Kühe, die gesund bleiben, werden fälschlicherweise als krank vorhergesagt. Das Multi-Layer-Perzeptron hingegen erkennt zwar mit 75% weniger als die SVM mit linearem Kernel der in naher Zukunft erkrankende Kühe als solche, dafür ist die Falsch-Positiv-Rate mit 28% deutlich geringer. Wenn das letztere Modell eingesetzt werden sollte, bleiben zwar unter Umständen einige Erkrankungen unerkannt, allerdings werden auch Kosten und Aufwand eingespart, da es weniger falsch-positive Alarme gibt.
Störungen der Trächtigkeit (ST) können insbesondere durch den Naive Bayes Klassifikator identifiziert werden. Sowohl vor als auch nach dem Undersampling wird diese Erkrankungsart zu 71% bzw 78% korrekt positiv klassifiziert. Allerdings werden bei dem balancierten Datensatz 58% der eigentlich gesund bleibenden Kühe als bald erkrankend klassifiziert, während beim unbalancierten Datensatz lediglich 39% falsch-positive Ergebnisse festzustellen sind. Störungen der Trächtigkeit werden sonst auch noch von der SVM mit rbf kernel beim balancierten Datensatz zu 65% korrekt positiv vorhergesagt, allerdings ist dort die Falsch-Negativ-Rate mit 34% um mehr als 10% höher als bei der Vorhersage durch Naive Bayes am balancierten Datensatz.
Eutererkrankungen (EU) bereiten die größten Schwierigkeiten in der Vorhersage. Ein Grund dafür ist womöglich, dass Eutererkrankungen in einem kleineren Umfang auftreten als die anderen beiden betrachteten Erkrankungsarten. Die SVM mit linearem Kernel klassifiziert etwa zwei Drittel der euter-erkrankenden Kühe korrekt als positiv, während ein Drittel dieser Kühe unerkannt bleibt. Sie ist damit das einzige Modell, das mehr als die Hälfte der in Zukunft an Euterkrankungen leidenden Kühe als solche korrekt vorhersagt.

Insgesamt lässt sich sagen, dass nur in wenigen Fällen mehr als drei Viertel der in naher Zukunft auftretenden Erkrankungen auch korrekt vorhergesagt werden. Durch Anpassen der Hyperparameter können die Vorhersagen vermutlich noch verbessert werden.

7.2 Herausforderungen

Die größte Herausforderung lag darin, eine geeignete Problemstellung zu finden, die anhand der Daten lösbar ist. Obwohl viele Krankheitsdaten vorlagen, war nur ein klei-

ner Teil davon nutzbar, weil ein Großteil davon bereits auftrat, bevor die erste Milchleistungsprüfung dokumentiert wurde. Die ursprüngliche Problemstellung musste angepasst werden, damit die Daten effektiv genutzt werden konnten.
Der geringe Teil an tatsächlich nutzbaren Krankheitsdaten führte außerdem zu einem stark unbalancierten Datensatz, der in dieser Form keine sinnvollen Vorhersagen hervorbrachte. Durch den Einsatz von Sampling Methoden konnte dies allerdings umgangen werden.
Eine weitere Herausforderung stellte die Integration des MLP-Datensatzes und des Krankheitsdatensatzes zu einem gemeinsamen Datensatz dar, weil dies zu einem großen Teil händisch erfolgen musste.

7.3 Vergleich mit Literatur

Versuche zur Vorhersage von Erkrankungen bei Milchkühen gibt es bereits (Slob, Catal und Kassahun, 2021). Diese wurden auch anhand von Daten über die Milchleistung bereits durchgeführt, allerdings wurden in diesen Arbeiten stets noch andere Datenquellen wie Infos über die Behausung, oder Daten aus medizinischen Untersuchungen wie Blutwerte und Hormonspiegel mit einbezogen (Tzelos et al., 2022; Lou et al., 2022). Der größte Unterschied zwischen dieser Arbeit und anderen Arbeiten ist jedoch, dass in anderen Untersuchungen die Daten direkt für eine spezielle Problemstellung erhoben wurde. So wird bspw. in Xu et al., 2019 zuerst der metabolische Zustand der Kuh anhand von Blutwerten bestimmt, dann werden die jeweiligen Daten der Milchleistungsprüfungen dem hinzu geordnet und danach versucht, den metabolischen Zustand der Kuh vorherzusagen, woraus ein mögliches Erkrankungsrisiko abgeleitet werden kann. In der vorliegenden Arbeit hingegen wird mit MLP-Daten gearbeitet, die routinemäßig aller ein bis zwei Monate erhoben werden. Diese wurden nicht mit dem Ziel erhoben, sie für Vorhersage von Erkrankungsrisiken zu nutzen. Ebenso wurden die Erkrankungsdaten nicht mit dem Ziel dokumentiert, sie für ein Machine Learning Projekt anzuwenden. Das hat zum einen zu Folge, dass mehr Datenvorverarbeitung betrieben werden musste. Zum anderen ist ein Großteil der Daten für die ursprüngliche Problemstellung nicht relevant und es mussten Kompromisse bei der Problemstellung eingegangen werden.

8 Fazit und Ausblick

Zusammenfassend lässt sich sagen, dass einige Modelle sinnvolle Vorhersagen über den allgemeinen Gesundheitszustand der Kuh sowie für einzelne Erkrankungsarten bieten. Dies war allerdings erst möglich, nachdem die ursprüngliche Problemstellung angepasst wurde. Da trotz dessen nur ein kleiner Teil der Daten genutzt werden kann, reduziert sich der Umfang der untersuchbaren Erkrankungsarten auf Erkrankungen des Bewegungsapparats (BW), Eutererkrankungen (EU) und Störungen der Trächtigkeit (ST). Außerdem wird in fast allen Fällen erst nach dem Balancieren des Datensatzes durch Undersampling eine sinnvolle Vorhersage erreicht, da zwischen den MLPs, denen eine Erkrankung zugeordnet wird und den MLPs, denen keine Erkrankung zugeordnet wird, ein zu großes Klassenungleichgewicht besteht. Die besten Ergebnisse können erzielt werden, wenn keine Informationen über vorherige MLPs einbezogen werden.
Um die Ergebnisse zu verbessern, sollten weitere Anpassungen bei den Hyperparametern vorgenommen werden. Beim Multi-Layer Perzeptron könnte die Anzahl der Neuronen im Hidden Layer oder die Aktivierungsfunktionen verändert werden. Beim Naive Bayes Klassifikator könnten die Wahrscheinlichkeiten der Attribute auf Grundlage einer anderen Verteilung geschätzt werden. Bei der Support Vector Machine könnten Parameter wie γ oder c verändert werden.
Darüber hinaus könnte die Performanz der Modelle an vollkommen neuen, unbekannten Daten überprüft werden. Alternativ kann getestet werden, ob die ursprüngliche Problemstellung durch mehr Daten lösbar wird. Im Anschluss daran könnte verglichen werden, ob die gleichen Modelle, die Erkrankungen vorhersagen können, die in naher Zukunft auftreten, auch Erkrankungen vorhersagen können, die in erst in der nächsten Laktation auftreten.

Originalarbeiten

Aggarwal, CC (2017) *Outlier Analysis*. Springer International Publishing. ISBN: 978-3-319-47577-6.

Ali, H et al. (2019) A review on data preprocessing methods for class imbalance problem 390–397.

Bauer, A, H Martens und C Thöne-Reineke (2021) Tierschutzrelevante Zuchtprobleme beim Milchvieh – Interaktion zwischen dem Zuchtziel „Milchleistung“ und dem vermehrten Auftreten von Produktionskrankheiten. *Berliner und Münchener Tierärztliche Wochenschrift* 134:1–9.

Bobbo, T et al. (2021) Comparison of machine learning methods to predict udder health status based on somatic cell counts in dairy cows. *Scientific Reports* 11(1).

Bzdok, D, N Altman und M Krzywinski (2018) Statistics versus machine learning. *Nature Methods* 15(4):233–234.

Chawla, N et al. (2002) SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res. (JAIR)* 16:321–357.

Chu, X et al. (2016) Data Cleaning: Overview and Emerging Challenges. *Proceedings of the 2016 International Conference on Management of Data*.

Clark, P und T Niblett (1989) The CN2 induction algorithm. *Machine Learning* 3(4):261–283.

Duan, K, S Keerthi und AN Poo (2003) Evaluation of simple performance measures for tuning SVM hyperparameters. *Neurocomputing* 51:41–59.

Duchi, J, E Hazan und Y Singer (2011) Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* 12:2121–2159.

Duda, R (1973) *Pattern classification and scene analysis*. Wiley. New York. ISBN: 978-0471223610.

Eisner, R et al. (2005). Improving Protein Function Prediction using the Hierarchical Structure of the Gene Ontology. In: *2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 1–10.

Friedman, N, D Geiger und M Goldszmidt (1997) Bayesian Network Classifiers. *Machine Learning* 29(2/3):131–163.

Galligan, D (2006) Economic Assessment of Animal Health Performance. *Veterinary Clinics of North America: Food Animal Practice* 22(1):207–227.

Ganganwar, V (2012) An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering* 2:42–47.

Garcia, S, J Luengo und F Herrera (2014) *Data Preprocessing in Data Mining*. Springer International Publishing.

Géron, A (2019) *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media. ISBN: 978-1-4920-3264-9.

Glatz-Hoppe, J, E Mohr und B Losand (2019) Nutzung von Milchinhaltsstoffen zur Beurteilung der Versorgungssituation von Milchkühen 2. Mitteilung: Bewertungsschema zur Beurteilung der Inhaltsstoffe auf Betriebsebene. *Züchtungskunde* 91(6):449–473.

Hand, DJ und K Yu (2001) Idiot's Bayes: Not So Stupid after All? *International Statistical Review / Revue Internationale de Statistique* 69(3):S.385.

Ilyas, IF und X Chu (2019) *Data Cleaning*. Association for Computing Machinery. New York, NY, USA. ISBN: 978-1-4503-7152-0.

Kingma, D und J Ba (2014) Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

Kohavi, R (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'95, 1137–1143.

KrishnaVeni, C und T Sobha Rani (2018). On the Classification of Imbalanced Data Sets. In: *International Journal of Computer Science and Technology.*

Kuhn, M, J Hutchison und H Norman (2007) Dry Period Length in US Jerseys: Characterization and Effects on Performance. *Journal of Dairy Science* 90(4):2069–2081.

Lou, W et al. (2022) Genetic analyses of blood -hydroxybutyrate predicted from milk infrared spectra and its association with longevity and female reproductive traits in Holstein cattle. *Journal of Dairy Science* 105(4):3269–3281.

Märtlbauer, E und H Becker (2016) *Milchkunde und Milchhygiene*. Ulmer. Stuttgart, Deutschland.

McCarthy, K, B Zabar und G Weiss (2005). Does Cost-Sensitive Learning Beat Sampling for Classifying Rare Classes? In: *Proceedings of the 1st international workshop on Utility-based data mining*. UBDM '05, 69–77.

McCulloch, WS und W Pitts (1943) A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics* 5(4):115–133.

Minsky, ML und S Papert (1969) *Perceptrons An Introduction to Computational Geometry*. Brand: MIT Press. ISBN: 978-0262130431.

Molinaro, AM, R Simon und RM Pfeiffer (2005) Prediction error estimation: a comparison of resampling methods. *Bioinformatics* 21(15):3301–3307.

Moran, J (2009) *Business Management for Tropical Dairy Farmers*. Landlinks Press.

Noble, WS (2006) What is a support vector machine? *Nature Biotechnology* 24(12):1565–1567.

Pedregosa, F et al. (2011) Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Polyak, B (1964) Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics* 4(5):1–17.

PraeRi (2020) Tiergesundheit, Hygiene und Biosicherheit in deutschen Milchkuhbetrieben - eine Prävalenzstudie (PraeRi). *Abschlussbericht 30.06.2020*.

Refaeilzadeh, P, L Tang und H Liu (2009) Cross-validation. *Encyclopedia of database systems* 5:532–538.

Rokach, L und O Maimon (2007) *Data Mining with Decision Trees*. World Scientific Publishing Company. ISBN: 978-981-4590-08-2.

Rosenblatt, F (1958) The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* 65(6):386–408.

Rumelhart, DE, GE Hinton und RJ Williams (1986). Learning internal representations by error propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, 318–362.

Shelke, MS, PR Deshmukh und VK Shandilya (2017). A Review on Imbalanced Data Handling Using Undersampling and Oversampling Technique. In: *International Journal of Recent Trends in Engineering Research (IJRTER)*, 444–449.

Slob, N, C Catal und A Kassahun (2021) Application of machine learning to improve dairy farm management: A systematic literature review. *Preventive Veterinary Medicine* 187:S.105237.

Sokolova, M und G Lapalme (2009) A systematic analysis of performance measures for classification tasks. *Information Processing Management* 45(4):427–437.

Strucken, EM, YCSM Laurenson und GA Brockmann (2015) Go with the flow - biology and genetics of the lactation cycle. *Frontiers in Genetics* 6.

Tharwat, A (2020) Classification assessment methods. *Applied Computing and Informatics* (ahead-of-print).

Tyagi, S und S Mittal (2020). Sampling Approaches for Imbalanced Data Classification Problem in Machine Learning. In: *Proceedings of ICRIC 2019*. Springer International Publishing, 209–221.

Tzelos, T et al. (2022) MiRNAs in milk can be used towards early prediction of mammary gland inflammation in cattle. *Scientific Reports* 12(1).

Van Hulse, J, T Khoshgoftaar und A Napolitano (2007). Experimental Perspectives on Learning from Imbalanced Data. In: *ACM International Conference Proceeding Series*, 935–942.

Xu, W et al. (2019) Prediction of metabolic status of dairy cows in early lactation with on-farm cow data and machine learning algorithms. *Journal of Dairy Science* 102(11):10186–10201.

Yang, Y (1999) An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval* 1(1/2):69–90.

Zhou, ZH (2021) *Machine Learning*. Springer Singapore. ISBN: 978-981-15-1966-6.

Webseiten

Alam, M. (2020) *Z-score for anomaly detection*. URL: https://towardsdatascience.com/z-score-for-anomaly-detection-d98b0006f510 (aufgerufen am 16. 07. 2022).

Eurostat. (2020) *Diagramm: Erzeugung von Kuhmilch*. URL: https://ec.europa.eu/eurostat/databrowser/product/view/APRO_MK_POBTA (aufgerufen am 28. 01. 2022).

Glatz-Hoppe, J, B Losand et al. (2020) *Milchkontrolldaten zur Fütterungs- und Gesundheitskontrolle bei Milchkühen. DLG Merkblatt 451*. URL: https://www.dlg.org/fileadmin/downloads/landwirtschaft/themen/publikationen/merkblaetter/dlgmerkblatt_451.pdf (aufgerufen am 10. 02. 2022).

LKV Sachsen. (2022) *Jahresabschluss GERO/MLP 2021*. URL: https://www.lkvsachsen.de/fileadmin/wireframe/redaktion/GERO_MLP/Jahresabschluss/2021/Jahresabschluss_GERO_MLP_2021.pdf (aufgerufen am 10. 02. 2022).

Martinek, V. (2020) *Cross-entropy for classification - Binary, multi-class and multi-label classification*. URL: https://towardsdatascience.com/cross-entropy-for-classification-d98e7f974451 (aufgerufen am 09. 09. 2022).

Pedregosa, F et al. (2011) *RBF SVM Parameters*. URL: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html (aufgerufen am 03. 08. 2022).

Ronaghan, S. (2018) *Deep Learning: Overview of Neurons and Activation Functions*. URL: https://srnghn.medium.com/deep-learning-overview-of-neurons-and-activation-functions-1d98286cf1e4 (aufgerufen am 03. 08. 2022).

Statistisches Bundesamt. (2021) *Pressemitteilung Nr. N 043: Viehhaltung im letzten Jahrzehnt: Weniger, aber größere Betriebe*. URL: https://www.destatis.de/DE/Presse/Pressemitteilungen/2021/07/PD21_N043_41.html (aufgerufen am 23. 02. 2022).

Yıldırım, S. (2020) *Hyperparameter Tuning for Support Vector Machines — C and Gamma Parameters*. URL: https://towardsdatascience.com/hyperparameter-tuning-for-support-vector-machines-c-and-gamma-parameters-6a5097416167 (aufgerufen am 27. 07. 2022).

Anhang

Parameter	Beschreibung
Betrieb	Nummer des Betriebs
MLP-Datum	Datum, an dem die MLP gemacht wurde
Ohrnummer	Ohrnummer zur Identifikation der Kuh
Kalbedatum	Datum, an dem die Kuh kalbte
Laktationstag	an welchem Tag der Laktation sich die Kuh zum Zeitpunkt der MLP befindet
Milchmenge in kg/Tag	abgegebene Milchmenge pro Tier und Tag
Fett in %	Fettanteil der Milch
Protein in %	Proteinanteil der Milch
Laktose in %	Laktoseanteil der Milch
Harnstoff in mg/L	Harnstoffgehalt in der Milch
pH-Wert	pH-Wert der Milch
Zellen in Tsd/mL	Anzahl der abgestorbenen Euterzellen in der Milch
C14-0 in %	Anteil der C14-0-Fettsäuren in der Milch
C16-0 in %	Anteil der C16-0-Fettsäuren in der Milch
C18-0 in %	Anteil der C18-0-Fettsäuren in der Milch
MilchC14-1 in %	Anteil der C14-1-Fettsäuren in der Milch
Saturated FA in %	Anteil der gesättigten Fettsäuren in der Milch
Total Unsaturated FA in %	Anteil aller ungesättigten Fettsäuren
Mono Unsaturated FA in %	Anteil aller Fettsäuren mit einer Doppelbindung
Poly Unsaturated FA in %	Anteil aller Fettsäuren mit mehreren Doppelbindungen
SCFA in %	Anteil der kurzkettigen Fettsäuren in der Milch
MCFA in %	Anteil der mittelkettigen Fettsäuren in der Milch
LCFA in %	Anteil der langkettigen Fettsäuren in der Milch
Trans FA in %	Anteil der Trans Fettsäuren in der Milch
De Novo FA-M in %	Anteil der DeNovo Fettsäuren
Mixed FA-M in %	Anteil der Mixed Fettsäuren
Preformed FA-M in %	Anteil der Preformed Fettsäuren

Tabelle A1: Primäre MLP-Daten.

Parameter	Beschreibung
FEQ (Fett-Eiweiß-Quotient)	$\frac{Fett}{Protein}$
Eiweiß-Energieverhältnis	$\frac{Protein \cdot 10}{0{,}95+0{,}21 \cdot Protein+0{,}38 \cdot Fett}$
Emax	$4,11 - 0,023 \cdot Milchmenge \cdot \frac{1+0{,}35}{3{,}51}$
Emin	$4,11 - 0,023 \cdot Milchmenge \cdot \frac{1-0{,}35}{3{,}51}$
Fmax	$5,06 - 0,033 \cdot Milchmenge \cdot \frac{1+0{,}68}{4{,}2}$
Fmin	$5,06 - 0,033 \cdot Milchmenge \cdot \frac{1-0{,}68}{4{,}2}$
De Novo FA-FA in % FA	Anteil der De Novo Fettsäuren von allen Fettsäuren
Mixed FA-FA in % FA	Anteil der Mixed Fettsäuren von allen Fettsäuren
Preformed FA-FA in % FA	Anteil der Preformed Fettsäuren von allen Fettsäuren
De Novo FA in g/Tag	Masse der De Novo Fettsäuren pro Tag
Mixed FA in g/Tag	Masse der Mixed Fettsäuren pro Tag
Preformed FA in g/Tag	Masse der Preformed Fettsäuren pro Tag

Tabelle A2: Sekundäre MLP-Daten.

Parameter	Beschreibung
Ohrnummer	Ohrnummer zur Identifikation der Kuh
Aktuelle Laktation	aktuelle Laktationsnummer, in der sich die Kuh befindet
Alter in Tagen	Alter der Kuh in Tagen
Datum (Geburt)	Geburtsdatum der Kuh
Datum (Abgang)	Abgangsdatum der Kuh
Grund (Abgang)	Grund des Abgangs
Alter (Abgang) in Tagen	Alter zum Zeitpunkt des Abgangs in Tagen
Laktationstag (Abgang)	Laktationstag, an dem sich die Kuh zum Zeitpunkt des Abgangs befand
Lebensleistung Milchmeng ein kg	Milchmenge in kg, die über gesamtes Leben geleistet wurde
Laktationsnummer	Laktationsnummer zum Zeitpunkt der Erkrankung
Datum Erkrankung	Datum, an dem die Erkrankung erfasst wurde
Art Erkrankung	Art der Erkrankung
Diagnose	Diagnose, die gestellt wurde
Klasse	Klasse der Erkrankung

Tabelle A3: Parameter, die im Datensatz über die Erkrankungen erfasst wurden.

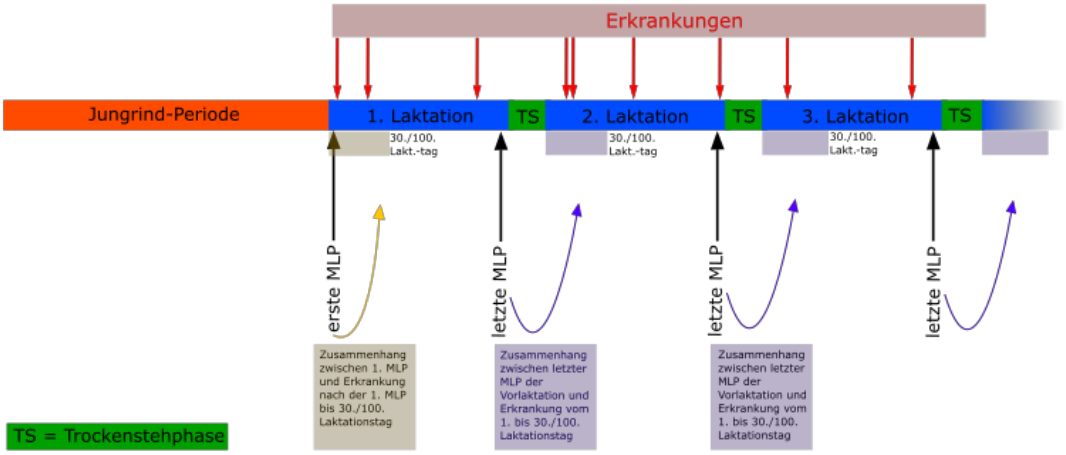


Abbildung A.1: Visualisierung der ursprünglichen Problemstellung. Die Abbildung wurde mit Inkscape Version 1.0 erstellt.

Klasse	Art der Erkrankung	Krankheit	n	n+1
SW1	Stoffwechsel (SW)	Ketose	ja	ja
SW2	Stoffwechsel (SW)	Prophylaxe	nein	nein
SW3	Stoffwechsel (SW)	Restliches (Gebär-/ Parese, Prophylaxe, Hypocalcämie, Allg.zustand, Labmagen, Indigestio,...)	ja	ja
BW1	Bewegungsapparat (BW)	Wandläsion, Klauensohlengeschwür, Dopp. Sohle, Klauenrehe, Kronsaumschwellung	ja	ja
BW2	Bewegungsapparat (BW)	Gelenkentzündung, Phlegmone, Arthritis, Ballhornfäule	ja	ja
BW3	Bewegungsapparat (BW)	RSG, Sohlen(spitzen)geschwür, Mortellaro, Limax	ja	ja
BW4	Bewegungsapparat (BW)	Trauma, Abzess	nein	nein
EU1	Euter (EU)	Mastitis	ja	ja
EU2	Euter (EU)	Verletzung	nein	nein
TU1	Trächtigkeitsuntersuchung (TU)	Anöstrie, Endometritis	ja	ja
TU2	Trächtigkeitsuntersuchung (TU)	Abort, Zysten	ja	ja
PA	Parasiten	Ektoparasiten	nein	nein
SE1	Infektionskrankheiten (SE)	Prophylaxe	nein	nein
SE2	Infektionskrankheiten (SE)	Restliches	nein	nein
SO1	Sonstige Krankheiten (SO)	Verdauung	ja	ja
SO2	Sonstige Krankheiten (SO)	Exitus, Pneumonie, Bronchitis, Fieber	nein	nein
ZH1	Fortpflanzungsstörung (ZH)	NG-beh, Abort, Endometritis	ja	ja
ZH2	Fortpflanzungsstörung (ZH)	Uterus	nein	nein
ZH3	Fortpflanzungsstörung (ZH)	Zysten, (Gelbkörper)	ja	ja
PK1	Puerperalkontrolle (PK)	Endometritis	eventuell	eventuell
PK2	Puerperalkontrolle (PK)	Zysten	eventuell	eventuell
ST	Störung Trächtigkeit	(Gelbkörper), Anöstrie, Follikel	ja	ja
KK	Kälberkrankheit	Otitis	nein	nein

Tabelle A4: Relevanz der unterschiedlichen Erkrankungsklassen innerhalb einer Laktation und über den Zeitraum einer Laktation hinaus.

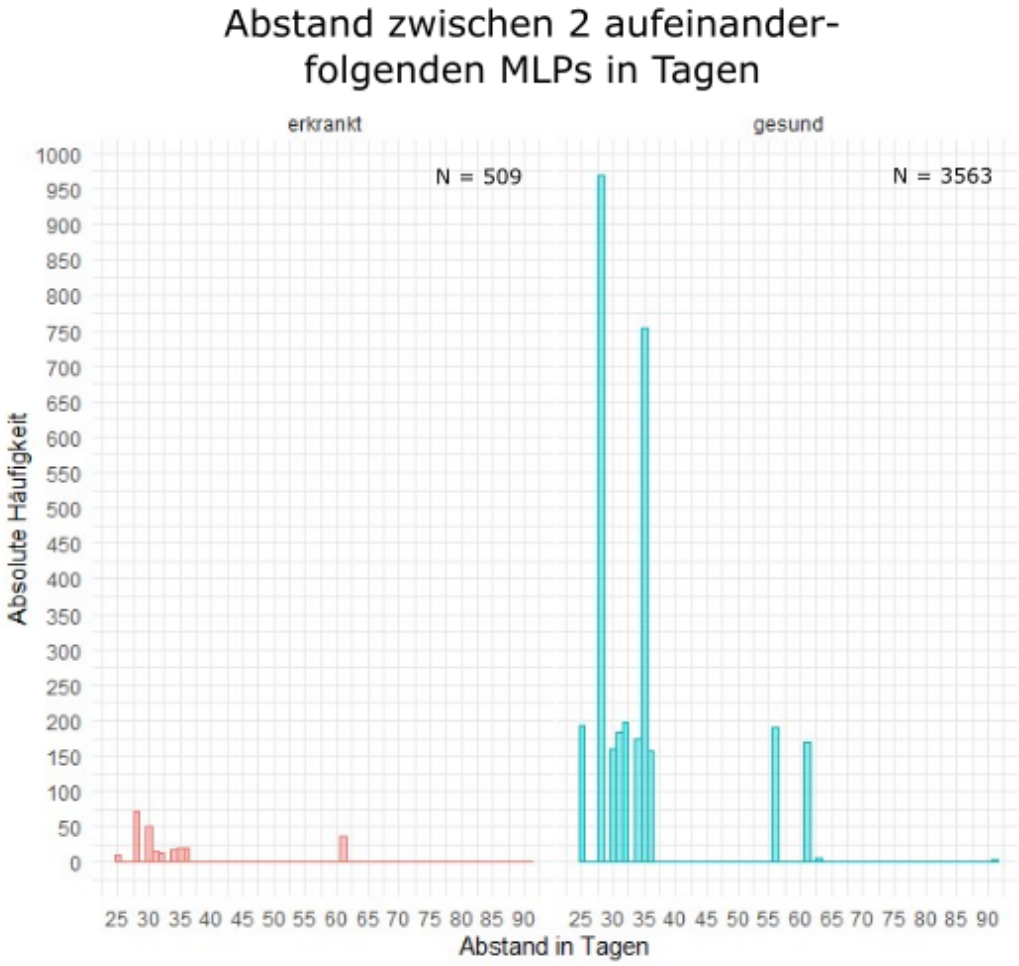


Abbildung A.2: **Abstand zwischen zwei aufeinander folgenden MLPs in Tagen.** Es wird unterschieden, ob im Zeitraum zwischen den beiden MLPs mindestens eine Erkrankung auftritt („erkrankt“) oder nicht („gesund“). Die Abbildung wurde mit dem R-package *ggplot2* erstellt und mit Inkscape V. 1.1.0 bearbeitet.

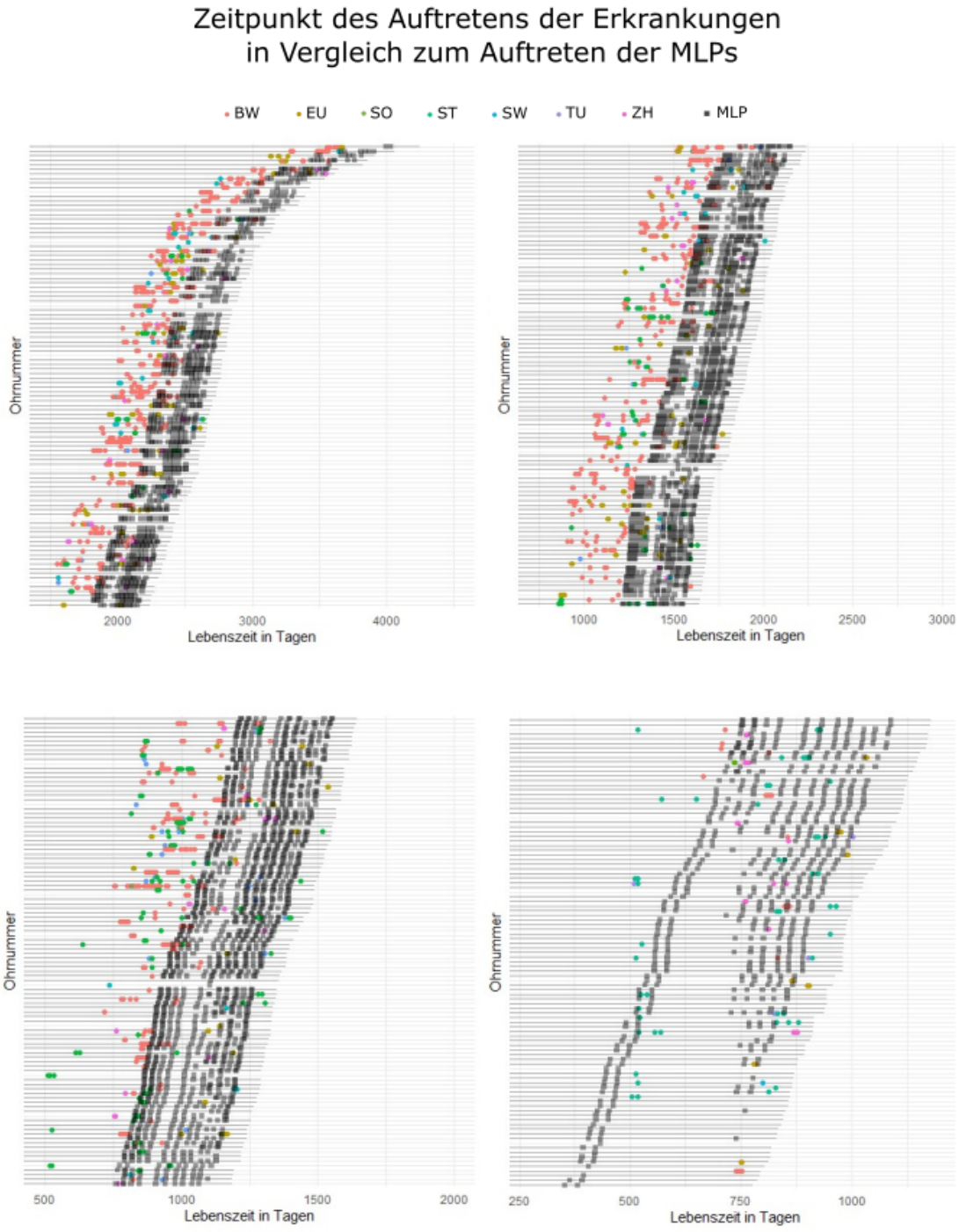


Abbildung A.3: **Zeitpunkt des Auftretens der Erkrankungen im Vergleich zum Zeitpunkt der MLPs.** Es wird zunächst die Lebensdauer der einzelnen Kühe als Linie dargestellt, auf der dann aufgetragen wird, wann eine MLP durchgeführt und wann eine Erkrankung festgestellt wurde. Die Abbildung wurde mit dem R-package *ggplot2* erstellt und mit Inkscape V. 1.1.0 bearbeitet.

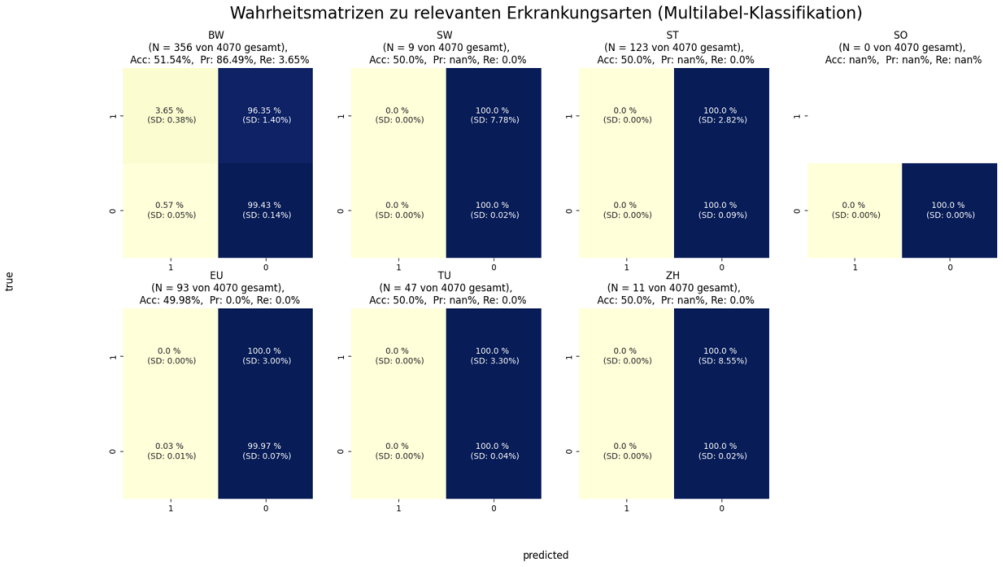


Abbildung A.4: Ergebnisse der Multilabel-Klassifikation des Multi-Layer Perzeptrons auf unveränderten, unbalancierten Daten.

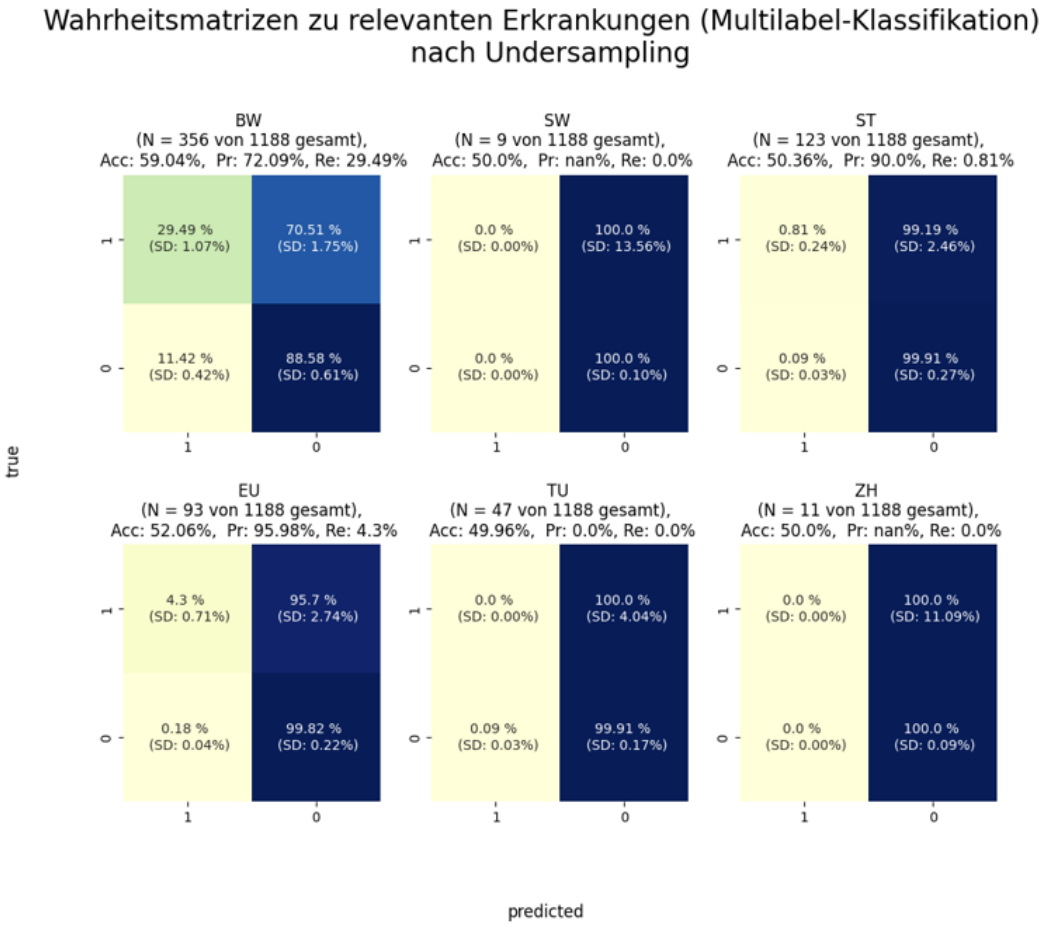
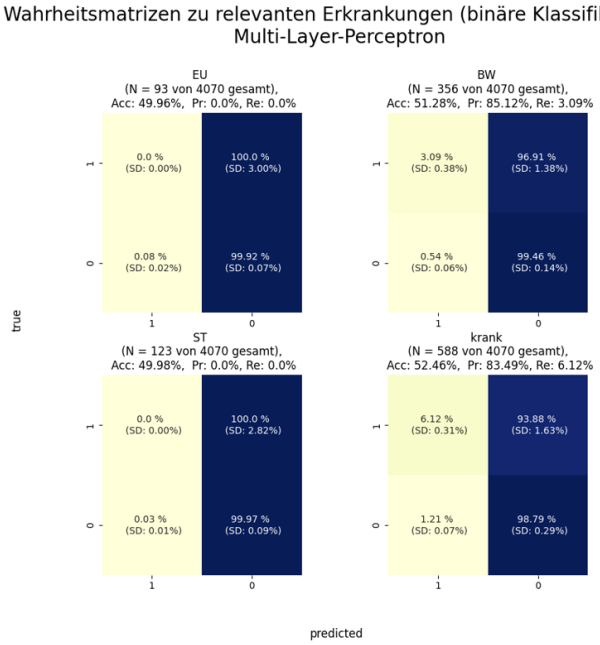
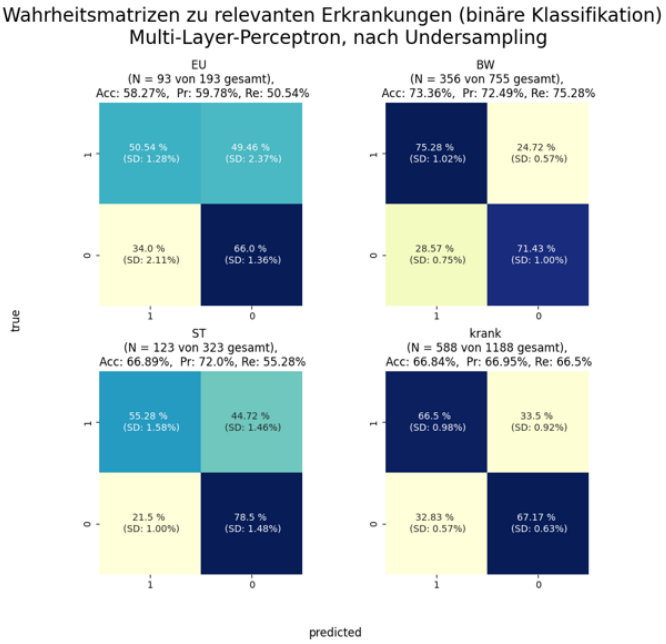


Abbildung A.5: Ergebnisse der Multilabel-Klassifikation des Multi-Layer Perzeptrons auf unveränderten, balancierten Daten.

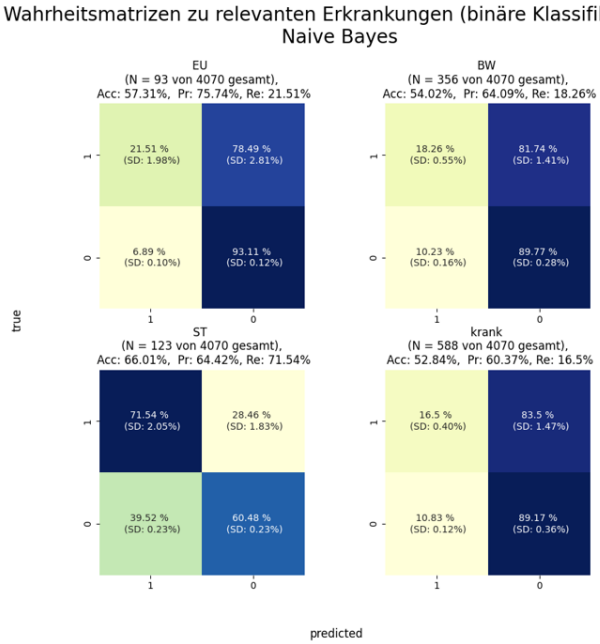


(a) Vorhersagergebnisse beim unbalancierten Datensatz.

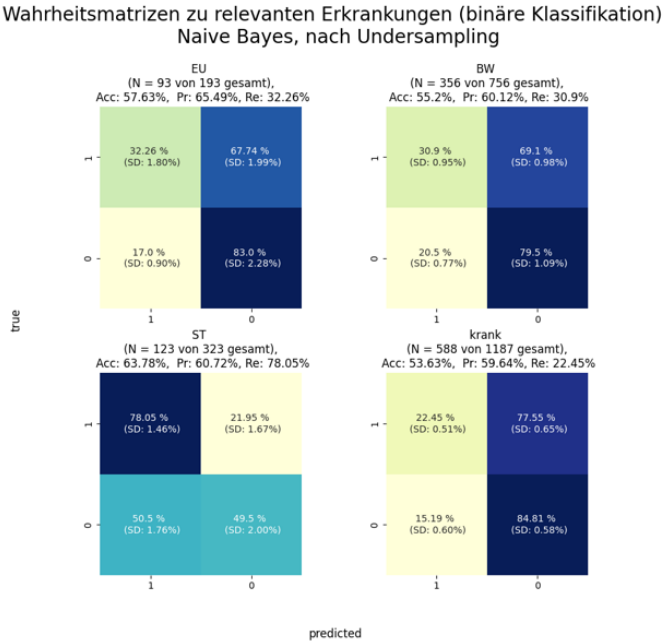


(b) Vorhersagergebnisse beim balancierten Datensatz.

Abbildung A.6: Ergebnisse der binären Klassifikation des Multi-Layer Perzeptrons auf unbalancierten und balancierten Daten.

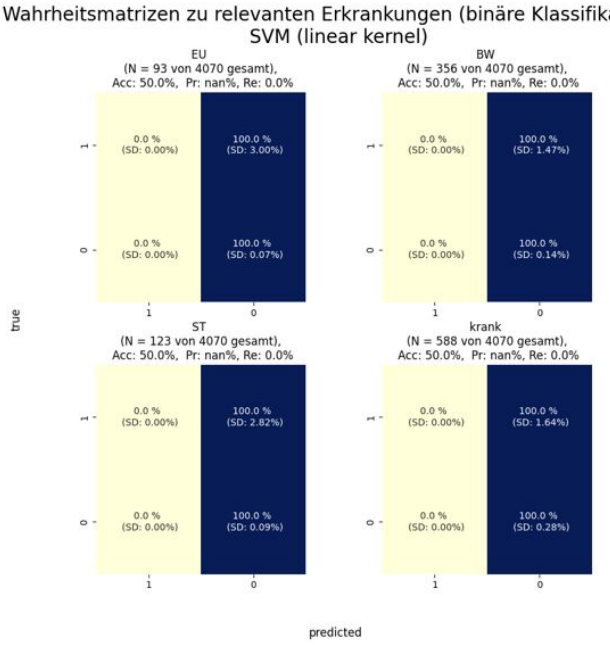


(a) Vorhersagergebnisse beim unbalancierten Datensatz.

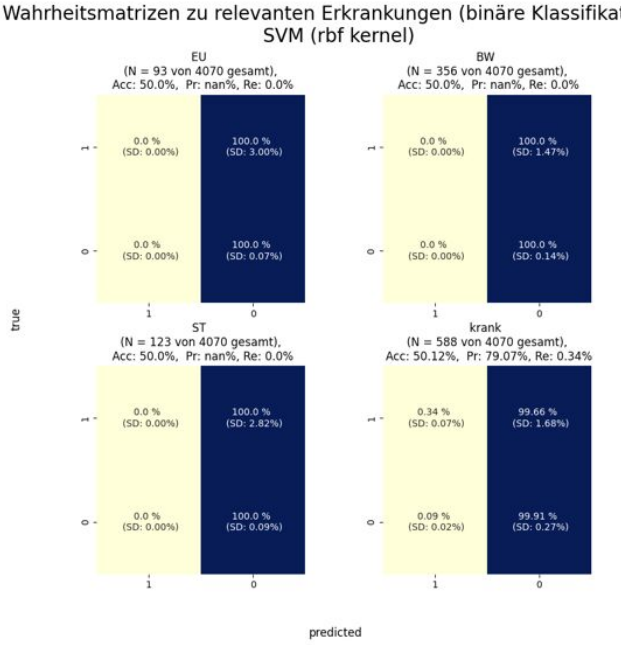


(b) Vorhersagergebnisse beim balancierten Datensatz.

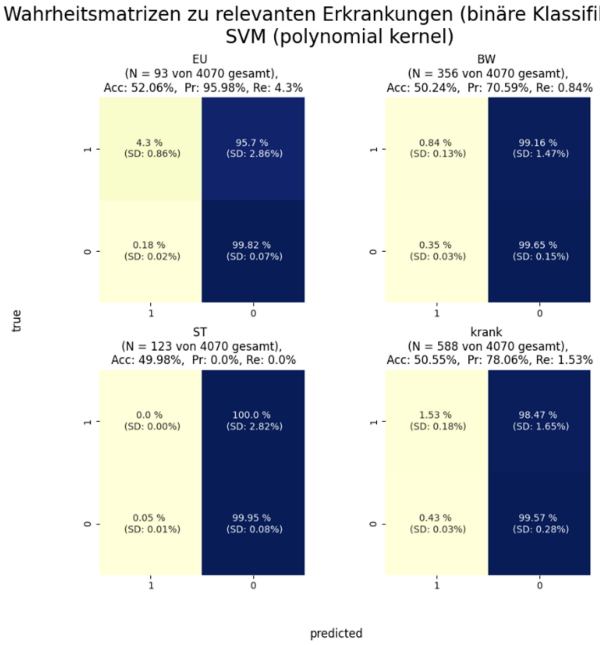
Abbildung A.7: Ergebnisse der binären Klassifikation durch Naive Bayes auf unbalancierten und balancierten Daten.



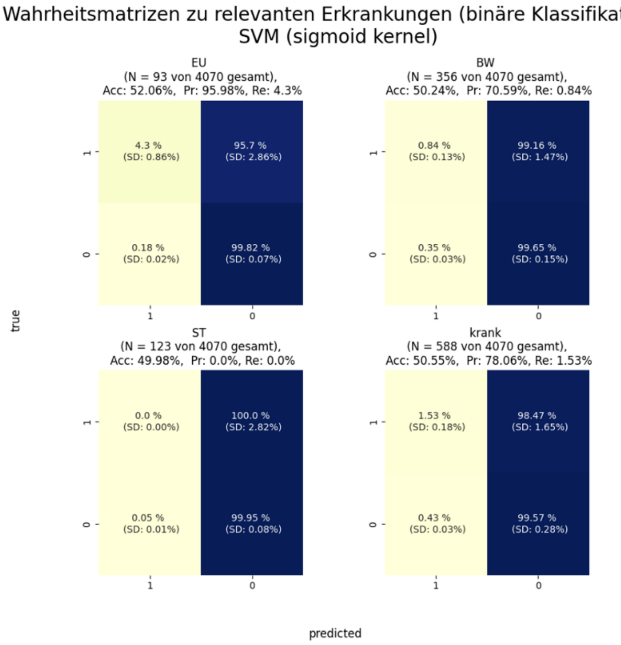
(a) Confusion Matrices der SVM mit linearem Kernel.



(b) Confusion Matrices der SVM mit rbf Kernel.

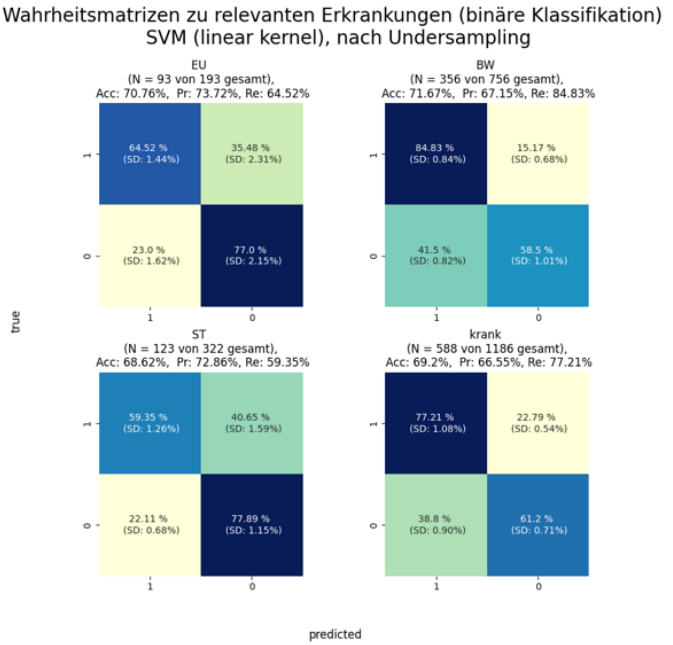


(c) Confusion Matrices der SVM mit polynomial Kernel.

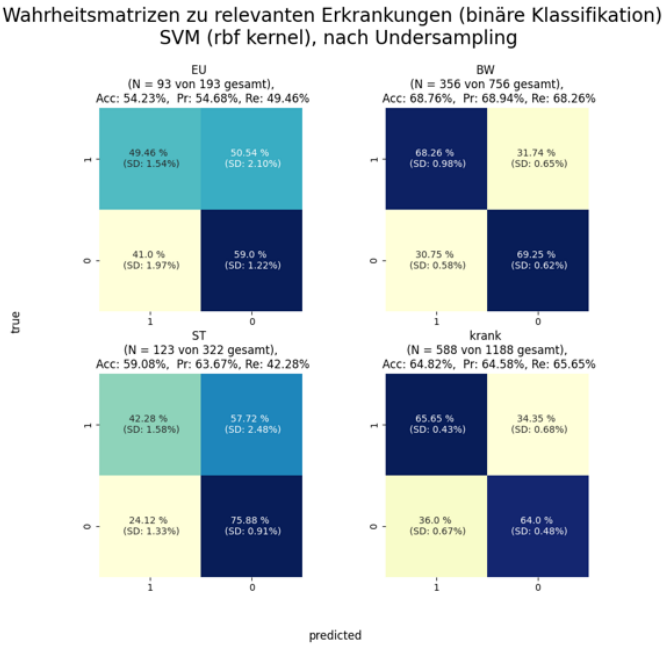


(d) Confusion Matrices der SVM mit sigmoid Kernel.

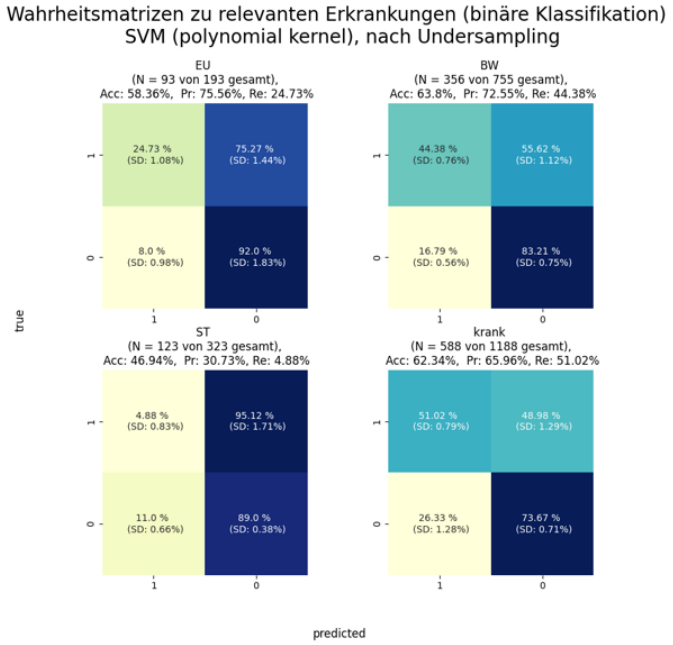
Abbildung A.8: Ergebnisse der binären Klassifikation durch SVMs mit verschiedenen Kernels. Die Modelle wurden auf dem unbalancierten Datensatz angewendet.



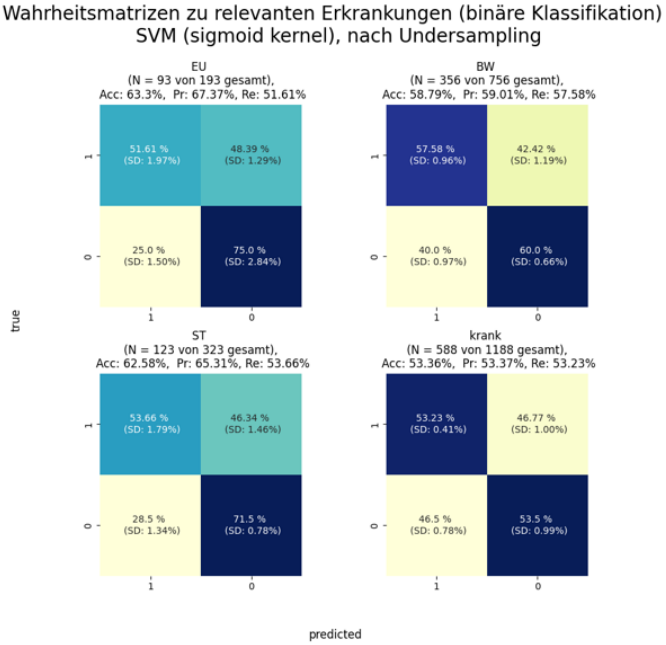
(a) Confusion Matrices der SVM mit linearem Kernel.



(b) Confusion Matrices der SVM mit rbf Kernel.



(c) Confusion Matrices der SVM mit polynomial Kernel.



(d) Confusion Matrices der SVM mit sigmoid Kernel.

Abbildung A.9: Ergebnisse der binären Klassifikation durch SVMs mit verschiedenen Kernels. Die Modelle wurden auf dem balancierten Datensatz angewendet.

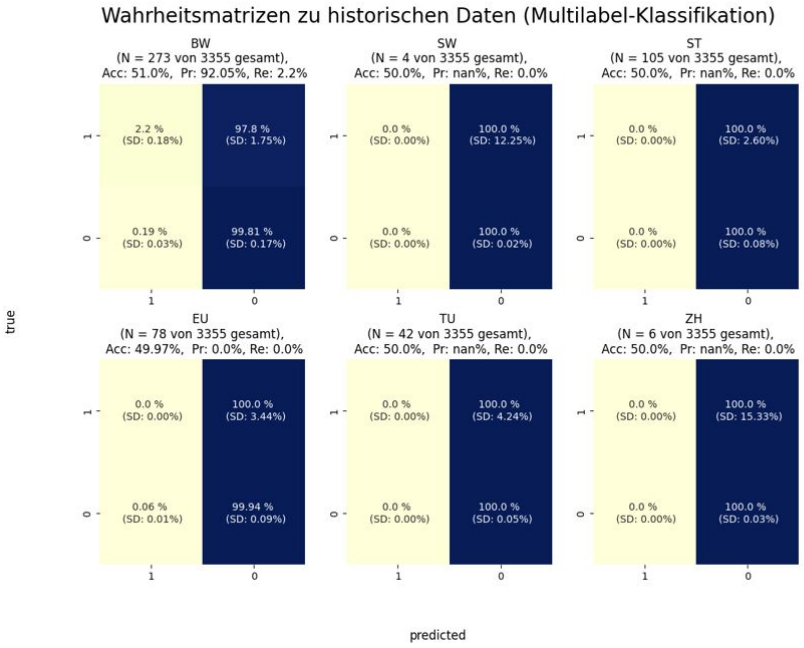


Abbildung A.10: Ergebnisse der Multilabel-Klassifikation des Multi-Layer Perzeptrons auf unbalancierten historischen MLP-Daten.

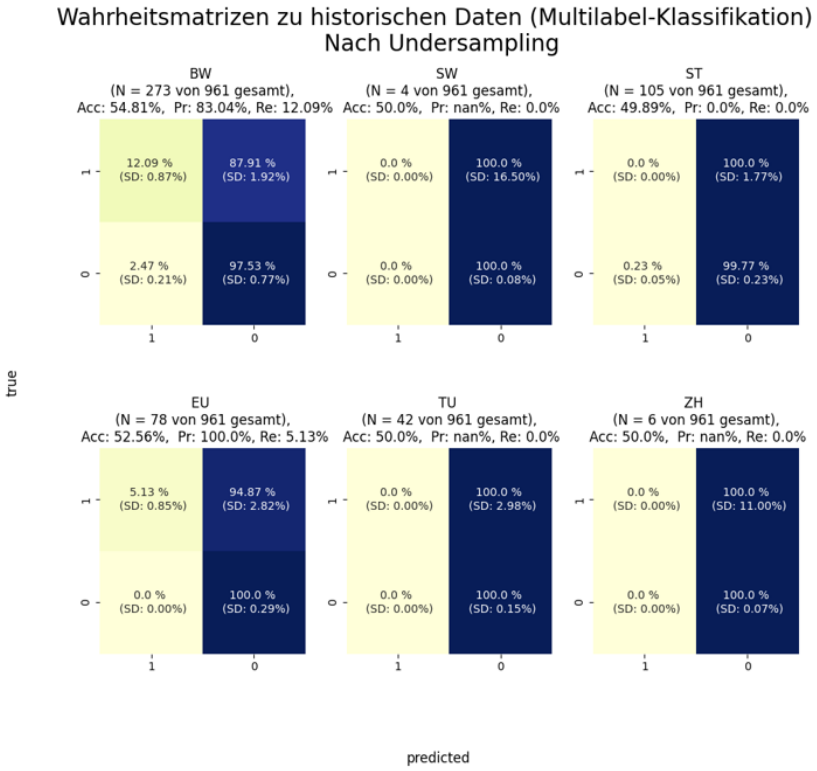
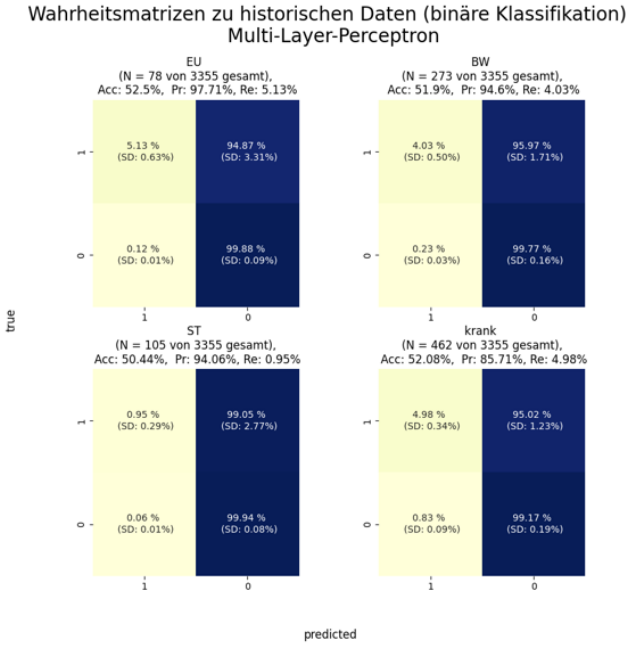
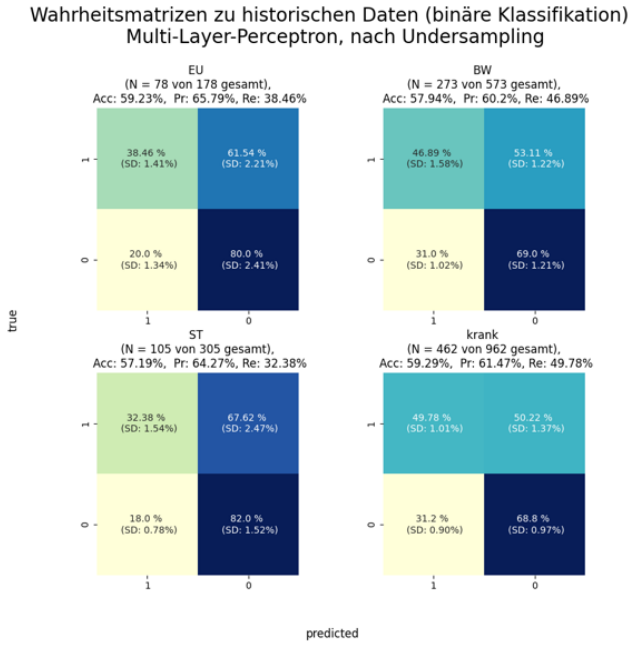


Abbildung A.11: Ergebnisse der Multilabel-Klassifikation des Multi-Layer Perzeptrons auf balancierten historischen MLP-Daten.

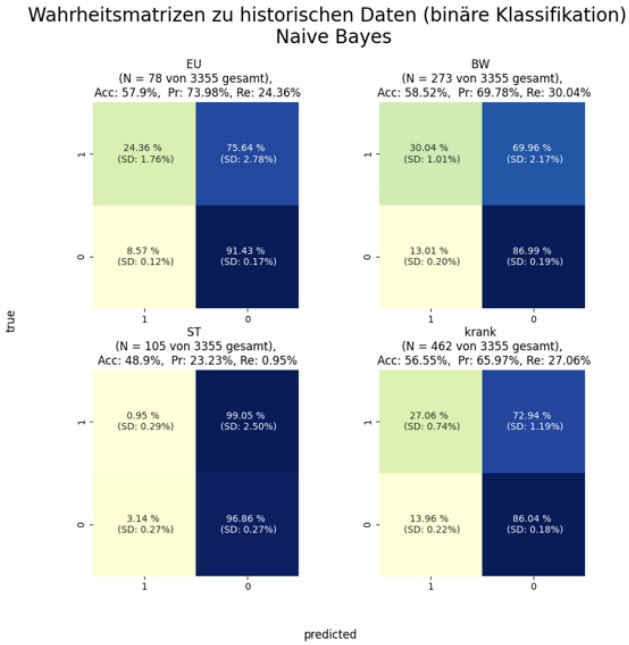


(a) Vorhersagergebnisse beim unbalancierten Datensatz.

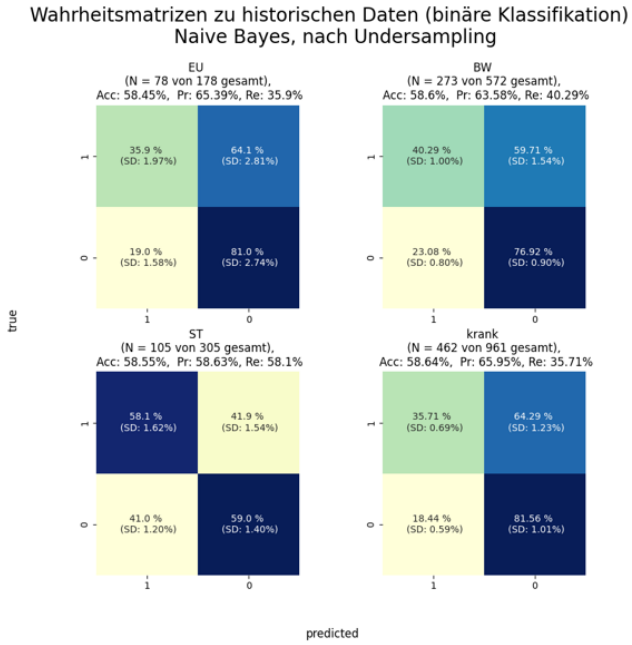


(b) Vorhersagergebnisse beim balancierten Datensatz.

Abbildung A.12: Ergebnisse der binären Klassifikation des Multi-Layer Perzeptrons auf unbalancierten und balancierten historischen MLP-Daten.

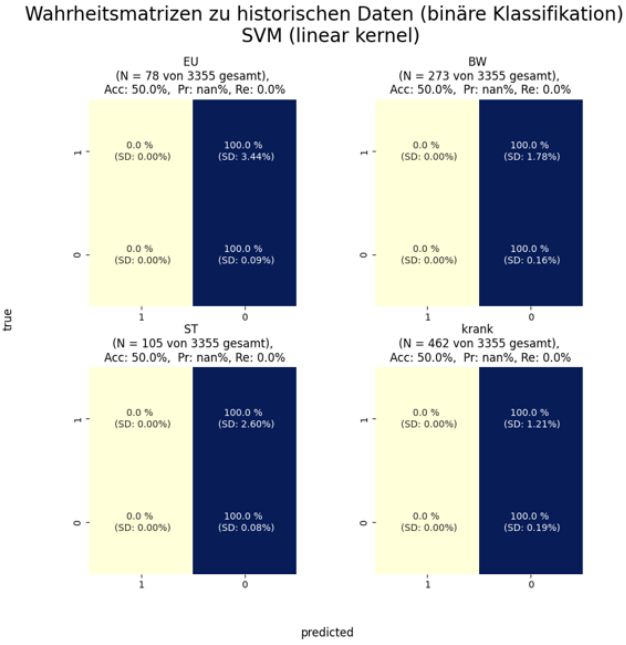


(a) Vorhersagergebnisse beim unbalancierten Datensatz.

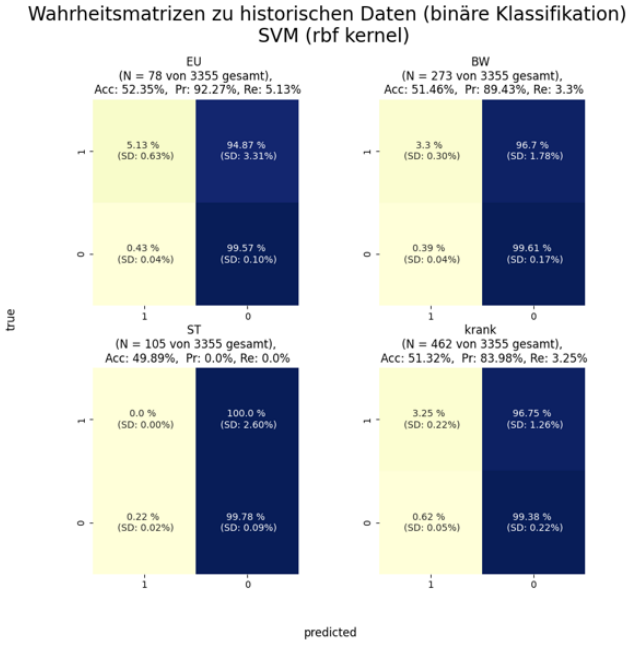


(b) Vorhersagergebnisse beim balancierten Datensatz.

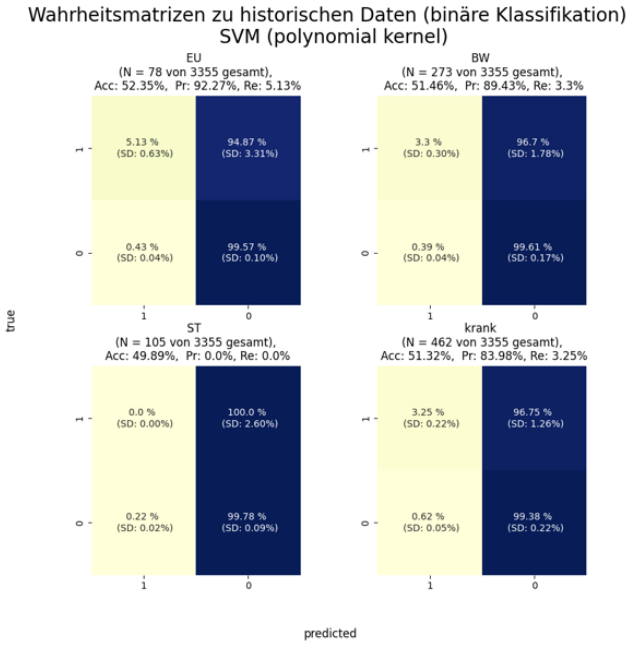
Abbildung A.13: Ergebnisse der binären Klassifikation durch Naive Bayes auf unbalancierten und balancierten historischen MLP-Daten.



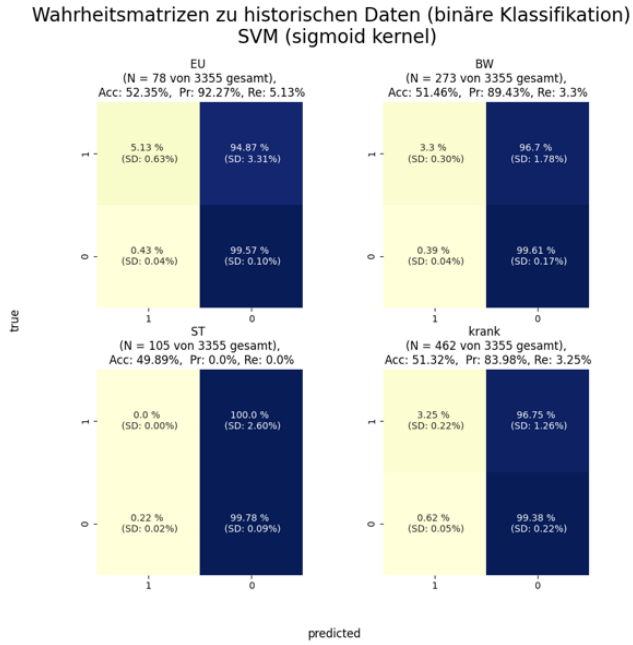
(a) Confusion Matrices der SVM mit linearem Kernel.



(b) Confusion Matrices der SVM mit rbf Kernel.

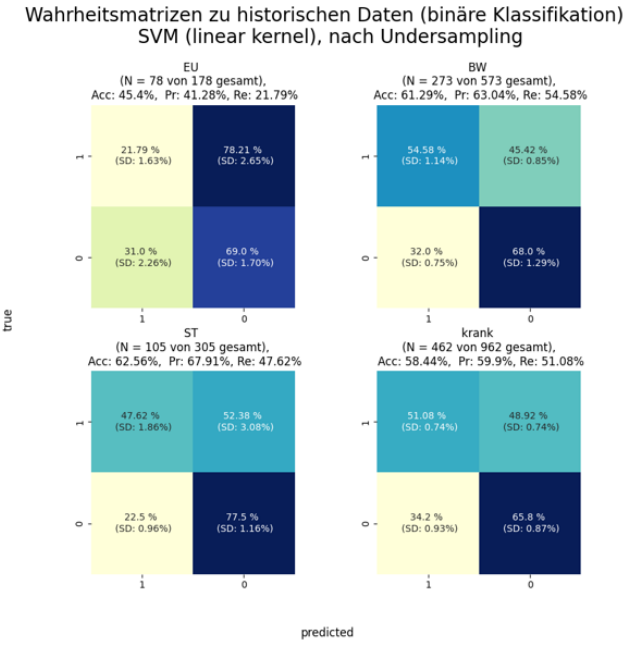


(c) Confusion Matrices der SVM mit polynomial Kernel.

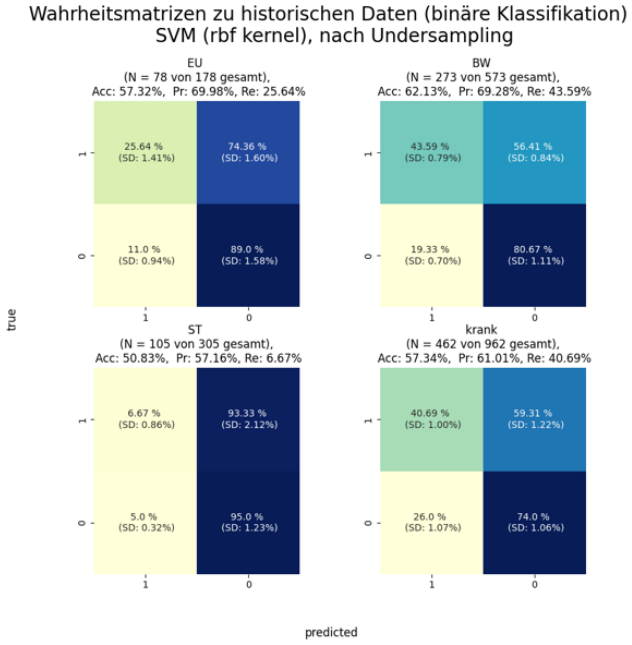


(d) Confusion Matrices der SVM mit sigmoid Kernel.

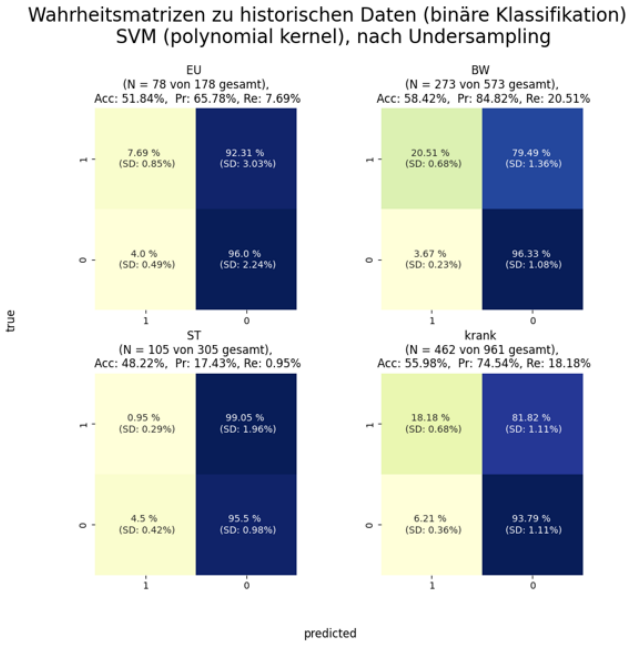
Abbildung A.14: Ergebnisse der binären Klassifikation durch SVMs mit verschiedenen Kernels. Die Modelle wurden auf dem unbalancierten Datensatz mit historischen MLP-Daten angewendet.



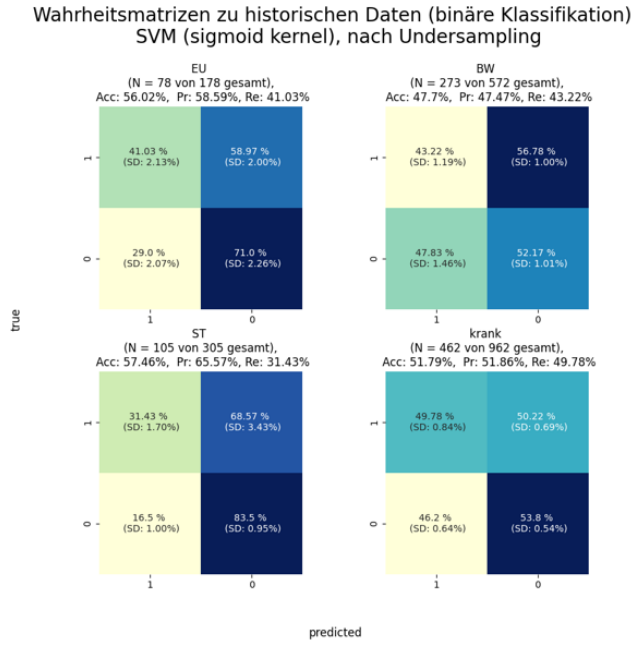
(a) Confusion Matrices der SVM mit linearem Kernel.



(b) Confusion Matrices der SVM mit rbf Kernel.



(c) Confusion Matrices der SVM mit polynomial Kernel.



(d) Confusion Matrices der SVM mit sigmoid Kernel.

Abbildung A.15: Ergebnisse der binären Klassifikation durch SVMs mit verschiedenen Kernels. Die Modelle wurden auf dem balancierten Datensatz mit historischen MLP-Daten angewendet.

Erklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Mittweida, 15.08.2022

Jasmin Kermer