



Universität Siegen

Projektseminar

Vorhersage der Unfallanzahl mit Personenschaden
in Mainz und Wiesbaden
mithilfe von
künstlichen neuronalen Netzen und polynomieller Regression

Gutachter

Dr. Viktor Bindewald

Prof. Dr. Marc Goerigk

Name:	Maurice Ahrens	Corinne Pretz
Matrikelnummer:	1600535	1454415
Email:	maurice.ahrens@student.uni-siegen.de	corinne.pretz@student.uni-siegen.de
Studiengang:	Business Analytics	Business Analytics
Abgabetermin:	08.07.2022	

Inhaltsverzeichnis

1 Einleitung	1
1.1 Zielsetzung	1
1.2 Aufbau der Arbeit	1
2 Grundlagen	2
2.1 Künstliche Intelligenz	2
2.1.1 Machine Learning	3
2.2 Polynomiale Regression	5
2.3 Künstliche neuronale Netze	7
3 Datensätze	11
3.1 Unfallatlas	11
3.1.1 Änderung des Datensatzes des Unfallatlases	14
3.2 Temperatur und Niederschlagshöhe	15
3.2.1 Änderung des Datensatzes der Temperatur und der Niederschlagshöhe	15
4 Implementierung und Anwendung	16
4.1 Vorhersage mithilfe einer polynomischen Regression	16
4.1.1 Grundmodell der polynomischen Regression	16
4.1.2 Erweitertes Modell der polynomischen Regression	18
4.2 Vorhersage mithilfe eines künstlichen neuronalen Netzes	24
4.2.1 Trainings-, Validierungs- und Testdaten	24
4.2.2 Aufbau des künstlichen neuronalen Netzes	25
4.2.3 Ergebnisse des künstlichen neuronalen Netzes	29

Abbildungsverzeichnis

1	Einordnung der Begriffe Künstliche Intelligenz, Machine Learning und Deep Learning nach [7].	2
2	Unterschied zwischen klassischen Algorithmen und Machine Learning Algorithmen [7].	3
3	Spezialfälle von Polynomfunktionen	5
4	Aufbau eines künstlichen neuronalen Netzes mit 2 Hidden-Layers.	7
5	Datenfluss eines künstlichen neuronalen Netzes [20].	8
6	Backpropagation Algorithmus	9
7	Entwicklung der Anzahl der Unfälle mit Personenschaden im Straßenverkehr in Deutschland über die Jahre 2016 bis 2020. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].	13
8	Anzahl der Unfälle mit Personenschaden in Deutschland, bezogen auf den Monat. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].	13
9	Anzahl der Unfälle mit Personenschaden in Deutschland bezogen auf den Wochentag. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].	14
10	Unfälle in Mainz im Monat	17
11	Regression für Mainz auf Monatsbasis	18
12	Unfälle in Mainz am Tag	19
13	Regression für Mainz auf Tagesbasis	20
14	Vorhersage für Mainz auf Tagesbasis	21
15	Neue Regression für Mainz auf Tagesbasis	22

16	Neue Vorhersage für Mainz auf Tagesbasis	23
17	Entwicklung des f1-Scores bei ansteigender Anzahl der künstlichen Neuronen im Hidden-Layer. Links zeigt den ersten Durchlauf für jeweils beide Städte, einmal mit Einbeziehen der Wetterdaten und einmal ohne. Die Plots rechts zeigen den zweiten Durchlauf.	27
18	Aufbau der künstlichen neuronalen Netze zur Vorhersage der Unfallanzahl mit Personenschaden zu einem bestimmten Zeitpunkt. Links für das Training mit Einbezug der Wetterdaten als Input. Rechts für das Training ohne Einbezug der Wetterdaten als Input. u gibt die maximale Anzahl an Unfällen an, die zu einem Zeitpunkt in der Stadt registriert worden sind, die als Datensatz genutzt wird.	28
19	Erster Durchlauf: F1-Score jeder Klasse bei Vorhersage bezüglich der Testdaten, nach dem Training mit den Daten der Stadt Mainz (oben) bzw. der Stadt Wiesbaden (unten). Links ohne, rechts mit Einbezug der Wetterdaten. . . .	31
20	Zweiter Durchlauf: F1-Score jeder Klasse bei Vorhersage bezüglich der Testdaten nach dem Training mit den Daten der Stadt Mainz (oben) bzw. der Stadt Wiesbaden (unten). Links ohne, rechts mit Einbezug der Wetterdaten.	32
21	Differenzen der vorhergesagten Unfallanzahlen zu den tatsächlichen Unfallanzahlen. Die Daten, für die die Vorhersage getätigt wird, sind alle vorhandenen Daten der jeweiligen Stadt (2016-2020). Die Vorhersagen werden mithilfe des künstlichen neuronalen Netzes getroffen, welches mit den Daten der jeweils anderen Stadt trainiert wurde	33

Tabellenverzeichnis

1	Dataframe für die monatlichen Unfälle in Mainz	16
2	Dataframe für die täglichen Unfälle in Mainz	19
3	Vorhersagen der polynomiellen Regression	23
4	Aufteilung der vorliegenden Daten in Trainings-, Validierungs- und Testdaten.	24
5	Klassenaufteilung der Unfallanzahlen zu einem Zeitpunkt in der Stadt Mainz.	25

6	Klassenaufteilung der Unfallanzahlen zu einem Zeitpunkt in der Stadt Wiesbaden.	25
7	Beispielhafte Vorhersage eines künstlichen neuronalen Netzes für die Stadt Mainz	29
8	F1-Scores der Vorhersage der Testdaten. Hierbei wird die Unfallanzahl der Testdaten mithilfe eines künstlichen neuronalen Netzes, welches mit den Trainingsdaten der jeweiligen Stadt trainiert wurde, vorhergesagt.	30
9	F1-Score und Accuracy der Vorhersage der Unfallanzahl einer Stadt mithilfe eines künstlichen neuronalen Netzes, welches mit den Trainingsdaten der jeweiligen anderen Stadt trainiert wurde.	33

1 Einleitung

Jährlich kommt es in Deutschland zu rund 150000 bis 200000 Unfällen, bei denen Personen zu Schaden kommen. Um zukünftig diese Zahlen zu reduzieren, kann es hilfreich sein, die Anzahl der Unfälle mit Personenschäden vorherzusagen. Somit können Zeitpunkte und Orte, an denen es häufig zu Unfällen kommt, lokalisiert und Maßnahmen ergriffen werden.

1.1 Zielsetzung

Ziel dieser Hausarbeit ist es, eine interaktive Anwendung zu entwickeln, welche dem Nutzer eine Vorhersage für Verkehrsunfälle mit Personenschäden für einen definierten Zeitpunkt ausgibt.

Dabei beschäftigt sich diese Arbeit mit der Entwicklung eines Vorhersagemodells zur Bestimmung der Unfallhäufigkeit mit Personenschäden in deutschen Städten. Für die Jahre 2016 bis 2020 stellen die statistischen Ämter des Bundes und der Länder Unfalldatensätze bereit. Diese Datensätze beinhalten jegliche Informationen zu Unfällen im Straßenverkehr, bei denen Personen zu Schaden gekommen sind. Zusätzlich soll der beschriebene Unfalldatensatz um weitere öffentlich verfügbare Informationen erweitert werden, um hierdurch die Qualität der Vorhersage zu verbessern. Dabei werden Daten genutzt, von denen erwartet wird, dass sie mit der Unfallhäufigkeit korreliert.

1.2 Aufbau der Arbeit

Diese Arbeit beginnt damit, in Kapitel 2 die benötigten Grundlagen, zu denen in dieser Arbeit genutzten Vorhersagemodellen, einzuführen. Dazu gehört der Begriff der künstlichen Intelligenz und dessen Teilgebiet, das Machine Learning. Zum einen wird in dieser Arbeit die polynomiale Regression und zum anderen künstliche neuronale Netze als Vorhersagemethode genutzt. Im folgenden Kapitel 3 wird der in dieser Arbeit verwendete Datensatz, der statistischen Ämter des Bundes und der Länder [22] beschrieben. Zudem wird erläutert, wie der Datensatz für die in dieser Arbeit genutzten Vorhersagemethoden entsprechend angepasst wird. Die Umsetzung der in Kapitel 2 erläuterten Vorhersagemethode wird in Kapitel 4 beschrieben und folgend die Ergebnisse der Vorhersagen analysiert und diskutiert. Somit soll eine Aussage darüber getroffen, ob die genutzten Vorhersagemethoden in diesem Bereich sinnvoll sind. Im 5. und letzten Kapitel werden die Ergebnisse noch einmal übersichtlich zusammengefasst und ein Ausblick in Erweiterungen der Vorhersagemethoden gegeben.

2 Grundlagen

In diesem Kapitel sollen die benötigten Grundlagen eingeführt und erläutert werden. Zum einen wird der Begriff der künstlichen Intelligenz beschrieben und eingeordnet. Damit einhergehend sollen die Grundlagen des maschinellen Lernens (engl. Machine Learning) eingeführt werden. Dabei wird auch auf das Teilgebiet der künstlichen neuronalen Netze eingegangen. Zusätzlich wird die polynomische Regression erläutert, welche in Kapitel 4.1 als Vorhersagemethode genutzt wird.

Der folgende Abschnitt 2.1 zur künstlichen Intelligenz ist aus der Seminararbeit „Probably Approximately Correct“ von Corinne Pretz entnommen [19].

2.1 Künstliche Intelligenz

Nach der Definition von E. Rich [21] ist die künstliche Intelligenz (KI, engl. Artificial Intelligence) das Bestreben, Computer dazu zu bringen, Dinge zu tun, in denen Menschen besser sind. Das maschinelle Lernen und das dabei oft verwendete Deep Learning sind Teilgebiete der künstlichen Intelligenz (siehe Abbildung 1).

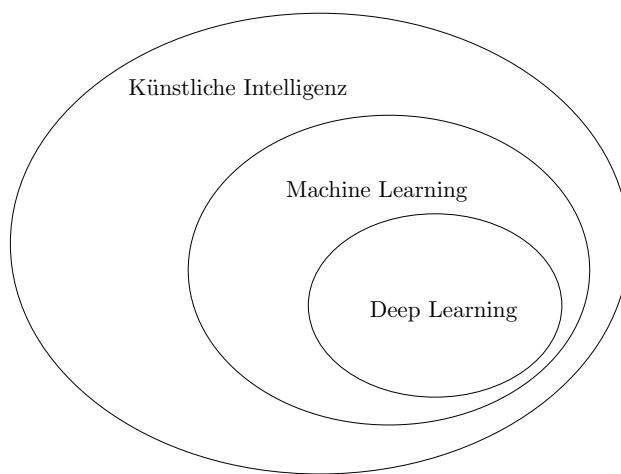


Abbildung 1: Einordnung der Begriffe Künstliche Intelligenz, Machine Learning und Deep Learning nach [7].

Auch ein Algorithmus, der nach den Regeln der klassischen Programmierung funktioniert, kann als intelligent bezeichnet werden und somit dem Gebiet der künstlichen Intelligenz zugeordnet werden. Ein Beispiel dafür ist ein Schach-Computer. Dieser wirkt nach außen sehr intelligent, obwohl dahinter kein Machine Learning Algorithmus steckt, sondern viele Regeln nach denen gerechnet wird. Dieser Bereich der künstlichen Intelligenz wird als symbolische

künstliche Intelligenz bezeichnet. Je komplexer und dynamischer Probleme jedoch werden, desto ungeeigneter ist die symbolische künstliche Intelligenz. Deshalb hat sich seit den 1980er Jahren der Ansatz des Machine Learning in vielen Forschungsgebieten durchgesetzt [2, 7, 8, 26].

2.1.1 Machine Learning

Als Machine Learning wird das Lernen von Computern anhand von Beispielen bezeichnet. Im Gegensatz zur klassischen Programmierung bekommt ein Machine Learning Algorithmus Daten und deren zugehörige Lösung als Input. Aus diesen Beispieldaten soll der Algorithmus schließlich die geltenden Regeln ermitteln, um diese auf zukünftige Daten anwenden zu können, deren Lösung nicht bekannt ist [7, 12]. Die unterschiedlichen Vorgehensweisen der klassischen Programmierung und des Machine Learning sind in Abbildung 2 dargestellt.

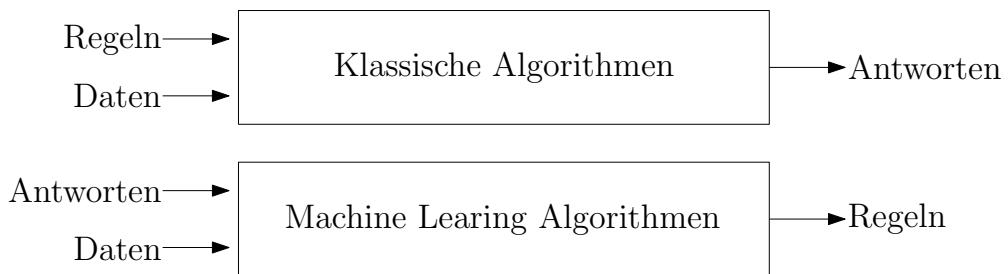


Abbildung 2: Unterschied zwischen klassischen Algorithmen und Machine Learning Algorithmen [7].

Damit die durch Machine Learning ermittelten Regeln auch auf zukünftige Daten angewendet werden können, wird zum Trainieren eine Vielzahl an Daten benötigt. Dies kann in der Praxis bei komplexen Problemen jedoch eine Herausforderung darstellen, da eine so große Menge an Daten womöglich nicht vorhanden ist oder aufwendig zu generieren ist. Es stellt sich zudem auch die Frage, wie viele Trainingsdaten benötigt werden beziehungsweise genügen, um ein Problem ausreichend zu lernen [7, 15].

Bewertungsmaß für Machine Learning

Um eine Aussage darüber zu treffen, wie gut ein Machine Learning Algorithmus funktioniert, gibt es unterschiedliche Bewertungsmaße.

Das wohl bekannteste und am häufigsten genutzte Bewertungsmaß ist die Accuracy. Die Accuracy (Gleichung 1) gibt das Verhältnis der vom Machine Learning Algorithmus richtigen Vorhersagen zur Gesamtzahl an Instanzen an.

$$A = \frac{\# \text{ richtig vorhergesagter Lösungen}}{\# \text{ der Lösungen insgesamt}} \quad (1)$$

Die Accuracy ist nur dann ein nützliches Bewertungsmaß, wenn die Klassen der Instanzen gleichmäßig verteilt sind und somit ein balanciertes Datenset vorliegt. Sobald ein unbalancedes Datenset vorliegt, in dem mehr Datenpunkte in einer Klasse als in einer anderen Klasse liegen, ist die Accuracy nicht besonders aussagekräftig.

Im Falle von unbalancierten Datensätzen kann der sogenannte f1-Score als Bewertungsmaß genutzt werden. Der f1-Score berücksichtigt dabei nicht nur die Anzahl der Vorhersagefehler, sondern auch die Art der Fehler, die gemacht werden. Der f1-Score setzt sich aus zwei Komponenten zusammen, der sogenannten Precision und dem Recall. Der Recall (Abdeckung, bei binären Klassifizierungen auch oft sensitivity) setzt sich dabei folgendermaßen zusammen:

$$\text{Recall} = \frac{\#\text{richtig positiv}}{\#\text{richtig positiv} + \#\text{falsch negativ}}.$$

Die Precision (Präzision) hingegen errechnet sich folgendermaßen [6]:

$$\text{Precision} = \frac{\#\text{richtig positiv}}{\#\text{richtig positiv} + \#\text{falsch positiv}}.$$

Der f1-Score setzt sich wie folgt aus Recall und Precision zusammen [14]:

$$\text{f1-Score} = 2 \cdot \frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}.$$

2.2 Polynomielle Regression

Eine Polynomfunktion ist eine Funktion, die aus der Summe von beliebig vielen Potenzfunktionen mit natürlichen Exponenten besteht. Diese Art der Funktion kann ausschließlich mittels der Operatoren Addition, Subtraktion und Multiplikation beschrieben werden [17].

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0.$$

Wobei $n \in N$ eine natürliche Zahl und $a_n, a_{n-1}, \dots, a_2, a_1, a_0 \in R$ reelle Zahlen sind und $a_n \neq 0$ gilt. Die Zahl n gibt den Grad der Funktion und die Zahlen $a_n, a_{n-1}, \dots, a_2, a_1, a_0$ dessen Koeffizienten an [3].

Polynomfunktionen werden auch ganzrationale Funktionen genannt, da sie zu der Klasse der rationalen Funktionen gehören. Bekannte Spezialfälle dieser Funktionen sind:

Lineare Funktion: $f(x) = a_1 x + a_0$.

Quadratische Funktion: $f(x) = a_2 x^2 + a_1 x + a_0$.

Kubische Funktion: $f(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$.

Quartische Funktion: $f(x) = a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x + a_0$.

Die grafische Darstellung dieser Spezialfunktionen sieht wie folgt aus:

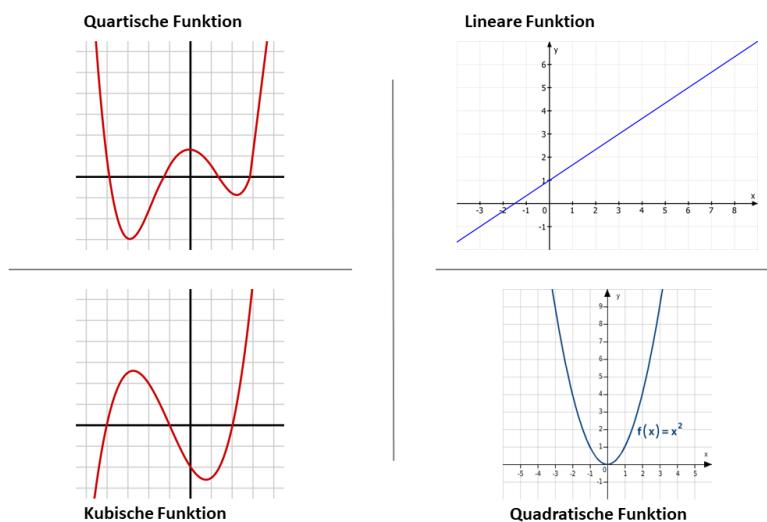


Abbildung 3: Spezialfälle von Polynomfunktionen

Es ist zu erkennen, dass je größer der Grad ist, den die Polynomfunktion annimmt, desto öfter tritt eine Änderung der Steigung auf. Diese Eigenschaft ist entscheidend für die Erstellung des Vorhersagemodells, da auch der Verlauf der Unfälle mit Personenschäden grafisch gesehen ständigen Steigungswechseln unterliegt. Somit ist es möglich eine Polynomfunktion zu erstellen, welche sich an den Verlauf der Unfälle mit Personenschäden anpasst und auf diese Weise die zukünftigen Unfallzahlen vorhersagt.

2.3 Künstliche neuronale Netze

Der folgende Abschnitt zum Aufbau und der Funktionsweise von künstlichen neuronalen Netzen ist aus der Bachelorarbeit von Corinne Pretz entnommen [18].

Künstliche neuronale Netze (KNN) sind ein Teilgebiet des Machine Learning. Ihr Aufbau orientiert sich an dem neuronalen Netz eines Lebewesens, jedoch sollen sie diese nicht imitieren. Ein künstliches neuronales Netz besteht aus einer oder mehreren Schichten (engl. Layers). Diese Schichten wiederum bestehen aus einer vom Anwender festgelegten Anzahl an künstlichen Neuronen. Diese künstlichen Neuronen der unterschiedlichen Schichten stehen in Wechselwirkung zu den künstlichen Neuronen der nächsten Schicht. Die erste Schicht eines künstlichen neuronalen Netzes wird als Input-Layer bezeichnet, die letzte als Output-Layer. Alle Layer zwischen dem Input- und Output-Layer heißen Hidden-Layer. Ein künstliches neuronales Netz mit einem Hidden-Layer wird dem Machine Learning zugeordnet. Besteht ein künstliches neuronales Netz jedoch aus mehr als nur einem Hidden-Layer, wird dieses dem Teilgebiet Deep Learning zugeordnet [1, 7, 16]. In der folgenden Abbildung 4 ist ein künstliches neuronales Netz mit 2 Hidden-Layers abgebildet.

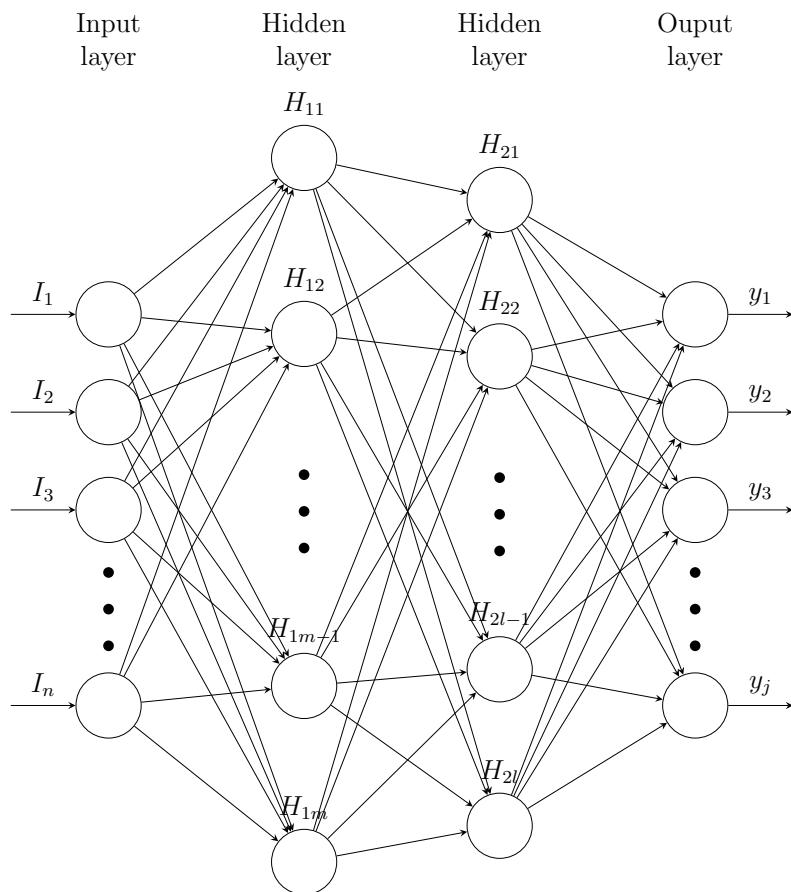


Abbildung 4: Aufbau eines künstlichen neuronalen Netzes mit 2 Hidden-Layers.

Datenfluss eines künstlichen neuronalen Netzes

Der Datenfluss eines künstlichen neuronalen Netzes beginnt mit dem Input Layer, welcher die Eingangsdaten in geeigneter Weise darstellt. Diese werden an den ersten Hidden-Layer weitergegeben und gehen in die Berechnung der sogenannten Aktivierung der Neuronen im ersten Hidden-Layer mit ein. Die Berechnung der Aktivierung der künstlichen Neuronen erfolgt folgendermaßen:

$$f(x_j) = f\left(\sum_{i \in I} w_i \cdot a_i + b\right) = y_j \quad \forall j \in J. \quad (2)$$

Die Menge J ist hierbei die Menge der künstlichen Neuronen in dem zu betrachtenden Layer, für welche die Aktivierung der Neuronen berechnet werden soll. I ist die Menge der Neuronen aus dem vorherigen Layer. $x_j = \sum_{i \in I} w_i \cdot a_i + b$ ist der gewichtete und um den Bias verschobene Input a_i des betrachteten künstlichen Neurons, wobei a_i zusätzlich die Output-Daten aus dem vorherigen Layer sind. w_i und b sind Parameter des künstlichen neuronalen Netzes. w_i stellen die Gewichte (engl. weights) und b den sogenannten Bias dar, welche beim Trainieren des künstlichen neuronalen Netzes mittels eines Optimierers ermittelt werden. Eine bildliche Darstellung des Datenflusses eines künstlichen Neurons ist in Abbildung 5 dargestellt.

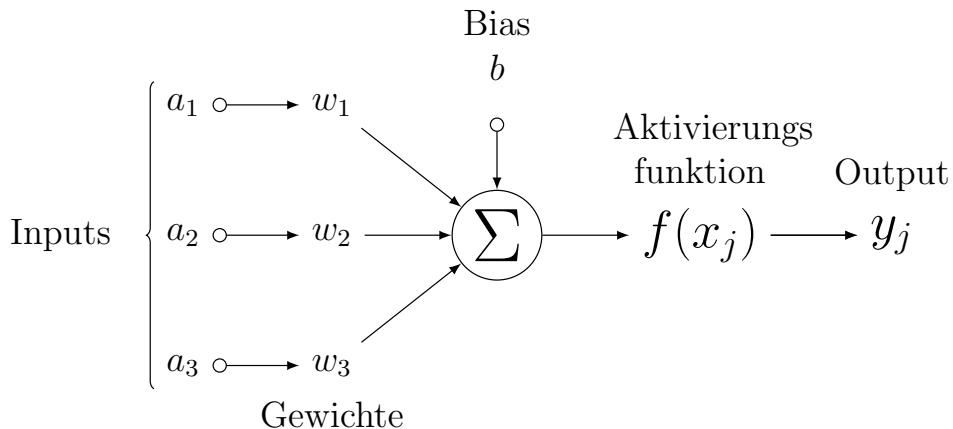


Abbildung 5: Datenfluss eines künstlichen neuronalen Netzes [20].

Der berechnete Wert x_j aus der Summe der Gewichte w_i , multipliziert mit den Daten a_i , addiert mit dem Bias b , wird in die sogenannte Aktivierungsfunktion eingesetzt. Der Output y_j ist die sogenannte Aktivierung der Neuronen und geht in die Berechnung der Aktivierung der Neuronen des nächsten Layers als Input a_i ein [1, 7, 23]. In den Referenzen [5, 25, 13, 24] sind unterschiedliche Aktivierungsfunktionen erläutert.

Trainingsalgorithmus

Damit ein künstliches neuronales Netz ein Problem lösen kann, muss es erst trainiert werden. Dazu werden Trainingsdaten und Validierungsdaten benötigt. Die Validierungsdaten werden zur Überwachung des Trainings genutzt und somit zur Anpassung der Hyperparameter (Anzahl der Hidden-Layer, Anzahl der künstlichen Neuronen in einem Hidden-Layer, etc.). Beide Datensätze beinhalten den Input des Problems und den Output, die Lösung. Mit diesen Daten kann das künstliche neuronale Netz durch Training die geltenden Regeln ermitteln (siehe Abbildung 2.1.1). Zum Training eines künstlichen neuronalen Netzes wird meist der sogenannte Backpropagation-Algorithmus genutzt, der in Abbildung 6 dargestellt ist [7].

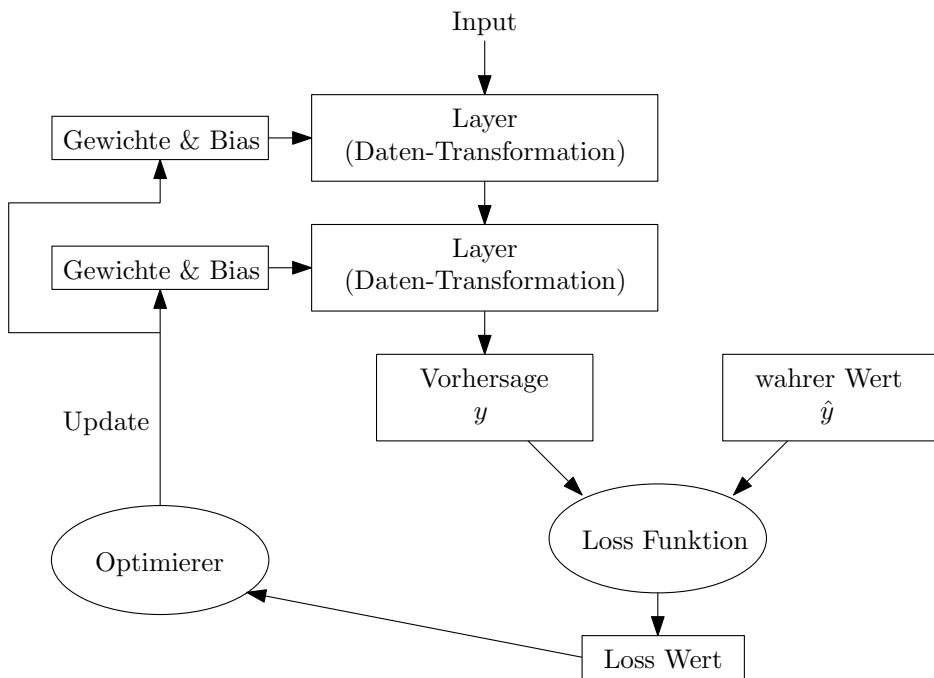


Abbildung 6: Backpropagation Algorithmus

Der erste Schritt ist das Übermitteln der Input-Daten an den ersten Hidden-Layer. In diesem werden mithilfe der zunächst zufällig gewählten Gewichte die Daten transformiert. Diese transformierten Daten, die mithilfe der Aktivierungsfunktion berechnet werden, gehen als Input in den nächsten Layer ein. Dort werden diese wieder mithilfe der Aktivierungsfunktion und der Gewichte transformiert. Dieser Vorgang setzt sich bis zum Output-Layer fort. Der Output-Layer gibt als Ausgabe eine vom künstlichen neuronalen Netz vorhergesagte Lösung zum Input des künstlichen neuronalen Netzes aus. Diese wird nun mithilfe einer Loss-Funktion mit der tatsächlichen Lösung aus dem Datensatz verglichen. Die Loss-Funktion gibt dabei an, wie stark die vorausgesagte Lösung des künstlichen neuronalen Netzes von der tatsächlichen Lösung des Problems abweicht. Ziel ist die Minimierung des Loss-Wertes.

Um den Loss zu minimieren, ermittelt ein Optimierer die Parameter, die Gewichte und den Bias so, dass der Loss im weiteren Training minimiert wird. Auch hier gibt es unterschiedliche Optimierer, die genutzt werden können. In den Referenz [11] sind mehrere Optimizer aufgelistet und erläutert. Mit diesen Informationen beginnt das künstliche neuronale Netz wieder bei den Input-Daten im ersten Layer und durchläuft alle Schritte bis zum Optimierer, der die Parameter wieder anpasst, bis möglichst alle Daten aus dem Trainings-Datenset „gelernt“ wurden [1, 7, 23, 25].

3 Datensätze

In diesem Kapitel werden die in dieser Arbeit genutzten Datensätze beschrieben. Zum einen werden die genutzten Datensätze der statistischen Ämter des Bundes und der Länder beschrieben und zum anderen die genutzten Datensätze des deutschen Wetterdienstes. Die statistischen Ämter des Bundes und der Länder veröffentlichen jährlich einen Datensatz, welcher die Informationen über alle polizeilich gemeldeten Straßenunfälle beinhaltet, bei welchen es zu Personenschäden gekommen ist. Es sind Datensätze der Jahre 2016 bis 2020 vorhanden. Diese dienen in dieser Arbeit als Hauptdatensatz zur Vorhersage von Unfallzahlen [22].

Zusätzlich soll ein weiterer Datensatz hinzugezogen werden, um die Voraussagen auf Grundlage des Hauptdatensatzes zu verbessern. Hierzu wird der Datensatz der Lufttemperatur und der Niederschlagshöhe des Deutschen Wetterdienstes für die Städte Mainz und Wiesbaden an den Wetterstationen Mainz-Lerchenberg (ZDF) und Wiesbaden-Auringen hinzugezogen. Die Datensätze der Lufttemperatur enthalten stündliche Daten der Lufttemperatur in Grad Celsius und der Niederschlagsdatensatz enthält stündliche Informationen zur Niederschlagshöhe in *mm*, an den Wetterstationen Mainz-Lerchenberg (ZDF) und Wiesbaden-Auringen. Beide Datensätze enthalten stündliche Daten von 01.05.2008 bis 27.05.2022 (Stand: 27.05.2022) [9, 10].

3.1 Unfallatlas

Die statistischen Ämter des Bundes und der Länder erfassen jährlich polizeilich aufgenommene Unfälle des Straßenverkehrs, bei welchen es zu Personenschäden oder Sachschäden gekommen ist. Der interaktive Unfallatlas der statistischen Ämter des Bundes und der Länder ermöglicht es, auf einer Deutschlandkarte Unfälle des Straßenverkehrs zu veranschaulichen, bei denen es zu Personenschäden gekommen ist. Dabei kann für einen gewählten Straßenabschnitt, die Anzahl an Unfällen mit Personenschäden abgelesen werden. Es liegen Daten der Jahre 2016 bis 2020 vor. Diese werden von den statistischen Ämtern des Bundes und der Länder als OpenData zur Verfügung gestellt. Die Datensätze erfassen alle in dieser Zeit polizeilich gemeldeten Straßenunfälle mit Personenschäden. Dabei werden folgende Informationen zu den Unfällen vermerkt:

- Unfall-ID
- amtlicher Gemeindeschlüssel der Stadt oder Gemeinde

- Zeitpunkt (Jahr, Monat, Wochentag, Stunde)
- Unfallkategorie
- Unfallart
- Unfalltyp
- Lichtverhältnisse
- Verkehrsmittel der Unfallbeteiligten
- Straßenzustand

Zum Zeitpunkt der Unfälle sind keine genauen Daten veröffentlicht. Lediglich der Monat, der Wochentag und die Stunde, in denen der Unfall passiert ist, sind veröffentlicht. Zusätzlich sind die Unfälle in Unfallkategorien eingeordnet. Dabei wird zwischen Unfällen mit getöteten, mit schwerverletzten und leichtverletzten unterschieden. Weiterhin ist die Art und der Typ des Unfalls dokumentiert. Unfallarten können dabei beispielsweise ein Zusammenstoß mit einem fahrenden Fahrzeug sein oder das Abkommen von der Fahrbahn. Unter Unfalltypen werden hier zum Beispiel ein Abbiegeunfall oder ein Unfall durch ruhenden Verkehr verstanden. Die Wetterverhältnisse zum Zeitpunkt des Unfalls werden ebenfalls angegeben. Dazu gehören die Lichtverhältnisse (Tageslicht, Dämmerung, Dunkelheit) und der Straßenzustand (trocken, nass, winterglatt). Neben der genauen Position des Unfalls werden außerdem die gewählten Verkehrsmittel der Unfallbeteiligten vermerkt. Die detaillierte Datensatzbeschreibung findet sich in Referenz [22].

Die Abbildungen 7, 8 und 9 veranschaulichen die Anzahl der Straßenunfälle in Deutschland. Abbildung 9 veranschaulicht dabei die Unfälle bezüglich des Wochentages, an dem der Unfall geschehen ist und Abbildung 7 und 8 bezüglich der Jahre.

Hierbei lässt sich in Abbildung 7 und 8 erkennen, dass die Anzahl an Unfällen seit 2016 ansteigen. Ausnahme ist dabei das Jahr 2020. Ein Grund hierfür könnte die Corona-Pandemie sein. Zum Anstieg der Unfallzahlen über die Jahre muss jedoch erwähnt werden, dass im Jahr 2016 die Unfalldaten von insgesamt sieben Bundesländer (Nordrhein-Westfalen, Niedersachsen, Mecklenburg-Vorpommern, Berlin, Brandenburg, Sachsen-Anhalt und Thüringen) nicht mit aufgefasst sind. Im Jahr 2017 und 2018 sind es nur noch drei Bundesländer (Nordrhein-Westfalen, Thüringen und Mecklenburg-Vorpommern) und im Jahr 2019 schließlich nur noch Mecklenburg-Vorpommern, die nicht in den Daten zu finden sind. Für das Jahr 2020 sind die Daten für ganz Deutschland gegeben.

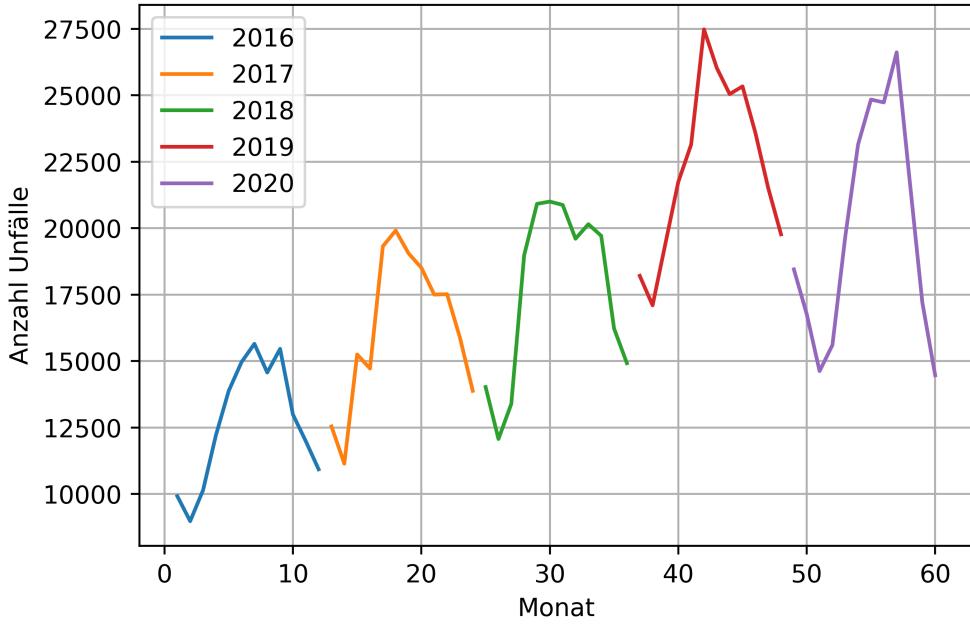


Abbildung 7: Entwicklung der Anzahl der Unfälle mit Personenschaden im Straßenverkehr in Deutschland über die Jahre 2016 bis 2020. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].

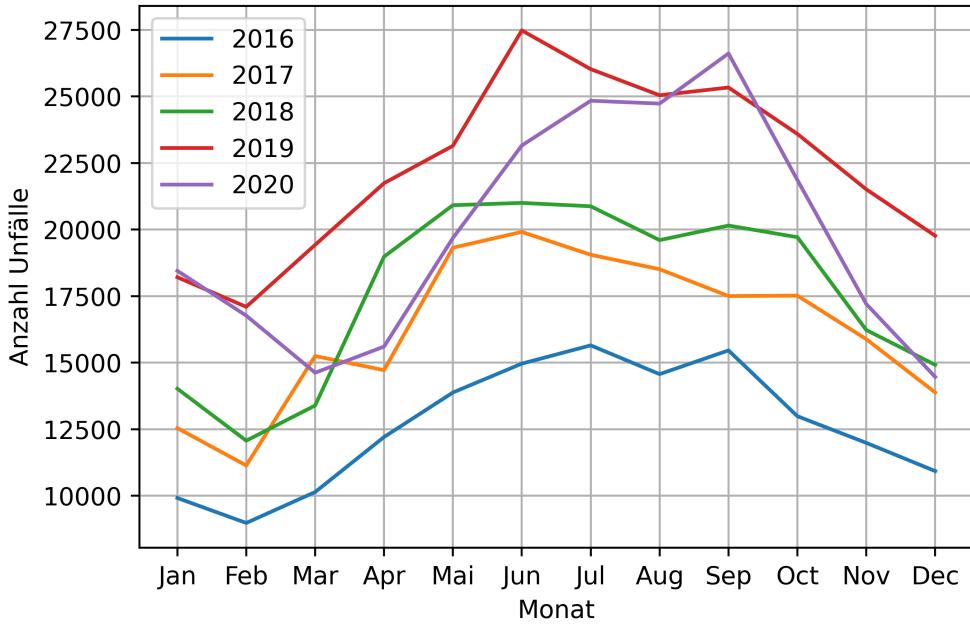


Abbildung 8: Anzahl der Unfälle mit Personenschaden in Deutschland, bezogen auf den Monat. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].

In Abbildung 9 lässt sich erkennen, dass unter der Woche (Montag bis Freitag) mehr Unfälle mit Personenschaden verursacht werden als am Wochenende. Besonders am Freitag steigen

die Zahlen in allen Jahren an. Das Wochenende zeigt dabei in alle Jahren die geringste Anzahl an Unfällen mit Personenschaden auf.

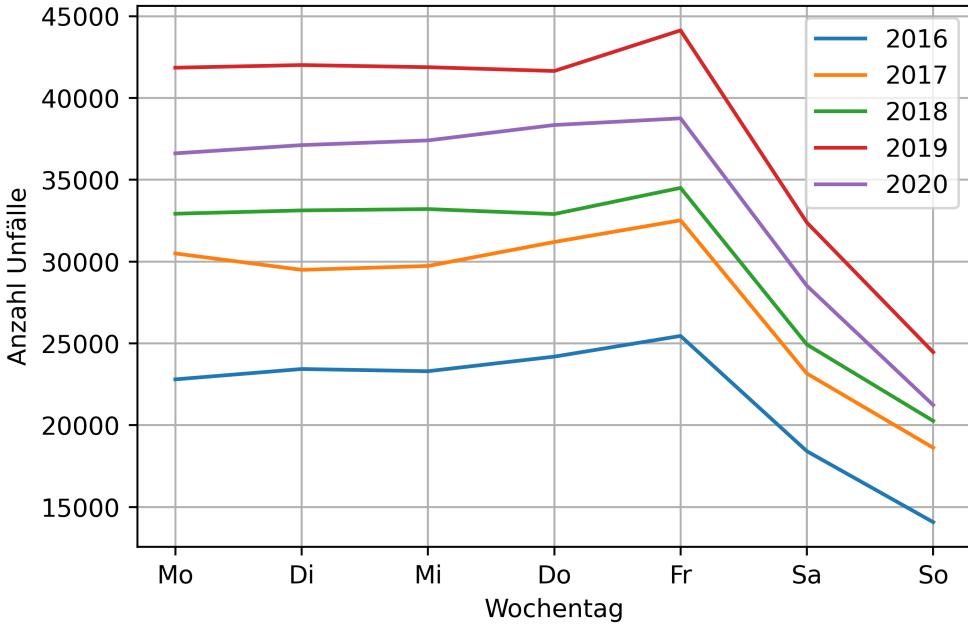


Abbildung 9: Anzahl der Unfälle mit Personenschaden in Deutschland bezogen auf den Wochentag. Hierbei fehlen teilweise Daten der im Text genannten Bundesländer [22].

3.1.1 Änderung des Datensatzes des Unfallatlases

Das Ziel dieser Arbeit ist es, eine Voraussage über die Unfallhäufigkeit zu einem bestimmten Zeitpunkt zu treffen. Da sich in dieser Arbeit die Vorhersage von Unfallzahlen auf den Raum Mainz und Wiesbaden beschränken soll, wird der Datensatz zuerst auf alle Daten dieser Städte reduziert. Dies wird anhand der amtlichen Gemeindeschlüssel der Städte gemacht. Da in dem hier genutzten Datensatz nur Informationen zu den Zeitpunkten enthalten sind, zu denen ein Unfall im Straßenverkehr mit Personenschaden entstanden sind, werden zur Vorhersage weitere Daten zu den Zeitpunkten benötigt, in denen kein Unfall mit Personenschaden verursacht worden ist. Hierzu wird der Datensatz um alle Zeitpunkte erweitert, die nicht im Datensatz vermerkt sind, da zu diesen Zeitpunkten kein Unfall mit Personenschaden entstanden ist, der polizeilich gemeldet wurde. Zusätzlich wird der Datensatz noch um die Angabe erweitert, wie viele Unfälle zu einem genannten Zeitpunkt geschehen sind. Ergänzend enthält der Datensatz einige Daten, die für die Vorhersage zur Unfallhäufigkeit unbrauchbar sind. Dazu gehören die Unfallart, der Unfalltyp, Unfall-ID, amtlicher Gemeindeschlüssel der Stadt oder Gemeinde, Unfallkategorie, Lichtverhältnisse, Verkehrsmittel der Unfallbeteiligten und der Straßenzustand. Somit ist der Datensatz vorzeitig auf die zeitliche

Angabe des Jahres, des Monats, der Stunde und der Unfallhäufigkeit beschränkt. Dies wird im Laufe dieser Arbeit als Input für das hier zu trainierende künstliche neuronale Netz und die polynomische Regression genutzt.

3.2 Temperatur und Niederschlagshöhe

Als weiterer Datensatz zur Verbesserung der Vorhersage wird unter anderem der Datensatz der Lufttemperatur der Städte Mainz und Wiesbaden genutzt. Diese Wahl wird getroffen, da anzunehmen ist, dass die Wettergegebenheiten mit der Unfallhäufigkeit korrelieren könnten. Die Temperaturdaten liegen in stündlichen Abständen für die jeweiligen Wetterstationen in 2 m Höhe und in Grad Celsius vor. Ebenfalls im Datensatz vermerkt ist die relative Luftfeuchte und das Qualitätsniveau der Daten. Zusätzlich zum Lufttemperatur-Datensatz wird der Datensatz der Niederschlagshöhe genutzt. Dieser enthält ebenfalls stündliche Daten von 01.05.2008 bis 27.05.2022 der Wetterstationen Mainz-Lerchenberg (ZDF) und Wiesbaden-Auringen. Hierbei setzt sich die stündliche Niederschlagshöhe aus sechs 10 min Messintervallen zusammen und wird in *mm* angegeben. Zusätzlich sind im Datensatz die Niederschlagsform (z. B. Tau oder gefallener Niederschlag), das Qualitätsniveau der Angaben und die binäre Angabe, ob Niederschlag gefallen ist oder nicht gegeben. Detailliertere Informationen zu den Datensätzen der Lufttemperatur und der Niederschlagshöhe finden sich in den Referenzen [9, 10]

3.2.1 Änderung des Datensatzes der Temperatur und der Niederschlagshöhe

In dieser Arbeit wird nur die stündliche Angabe, der Niederschlagshöhe und der Lufttemperatur der Jahre 2016 bis 2020 benötigt. Somit wird der Datensatz auf diese Zeitspanne verringert und dem Unfalldatensatz zeitgenau hinzugefügt. Hierzu werden Mittelwerte der einzelnen Wochentage in einem Monat berechnet, da zu den Unfällen keine genauen Daten bekannt sind. Der Datensatz zur Lufttemperatur wird weitergehend um die angegebene relative Feuchte und verringert. Der Datensatz zur Niederschlagshöhe wird um die Niederschlagsform und die binäre Angabe reduziert.

4 Implementierung und Anwendung

In diesem Kapitel wird die Umsetzung der in Kapitel 2 beschriebenen Vorhersagemethoden auf den in Kapitel 3 beschriebenen Datensatz erläutert. Zum einen wird die Anwendung der polynomiellen erläutert und beschrieben und zum anderen wird auf den Aufbau des genutzten künstlichen neuronalen Netzes eingegangen. Bezüglich der Vorhersage mittels eines künstlichen neuronalen Netzes wird ebenfalls auf die Nutzung der Aktivierungsfunktionen, genutzter Optimizer, Loss - und Genauigkeits - Berechnung und Ausgabedeutung des künstlichen neuronalen Netzes eingegangen. Im Anschluss werden die Ergebnisse beider Vorhersagemethoden erläutert.

4.1 Vorhersage mithilfe einer polynomiellen Regression

In diesem Abschnitt geht es um die Entwicklung und Analyse der Leistung der polynomiellen Regression zur Vorhersage der Unfallzahlen mit Personenschaden.

4.1.1 Grundmodell der polynomiellen Regression

Eine große Schwäche an dem Datensatz des Unfallatlas der statistischen Ämter des Bundes und der Länder ist, dass nicht hervorgeht, an welchem Tag sich genau ein Unfall mit Personenschaden ereignet hat. Zudem können Modelle in der Theorie zwar vielversprechend klingen, jedoch an verschiedenen Daten komplett unterschiedlich gut performen. Aus diesen beiden Gründen haben wir uns dazu entschieden, die polynomielle Regression zunächst für die Vorhersage für die Unfallzahlen in der Stadt Mainz auf monatlicher Basis zu testen. Für dieses Vorhaben wurde zunächst der Datensatz des Unfallatlas eingelesen und im Anschluss umstrukturiert. Zu Beginn wurde der Dataframe auf die Informationen aus der Stadt Mainz gefiltert. Die polynomielle Regression benötigt als Input eine Zeitreihe, an welcher sich die Regressionskurve anpassen kann. Aus diesem Grund wurde der Dataframe im nächsten Schritt so bearbeitet, dass die Monate für das Jahr 2016 und die Unfallzahlen im jeweiligen Monat ausgegeben werden.

Monat	1	2	3	4	5	6	7	8	9	10	11	12
Anzahl Unfälle	38	50	46	60	58	76	75	62	65	63	64	59

Tabelle 1: Dataframe für die monatlichen Unfälle in Mainz

Die Vorbereitung des Dataframes für den Test der polynomiellen Regression ist somit abgeschlossen. Damit ersichtlich ist, wie sich die Regressionskurve auf die Daten anpasst, wird der Dataframe zusätzlich in einem Plot veranschaulicht.

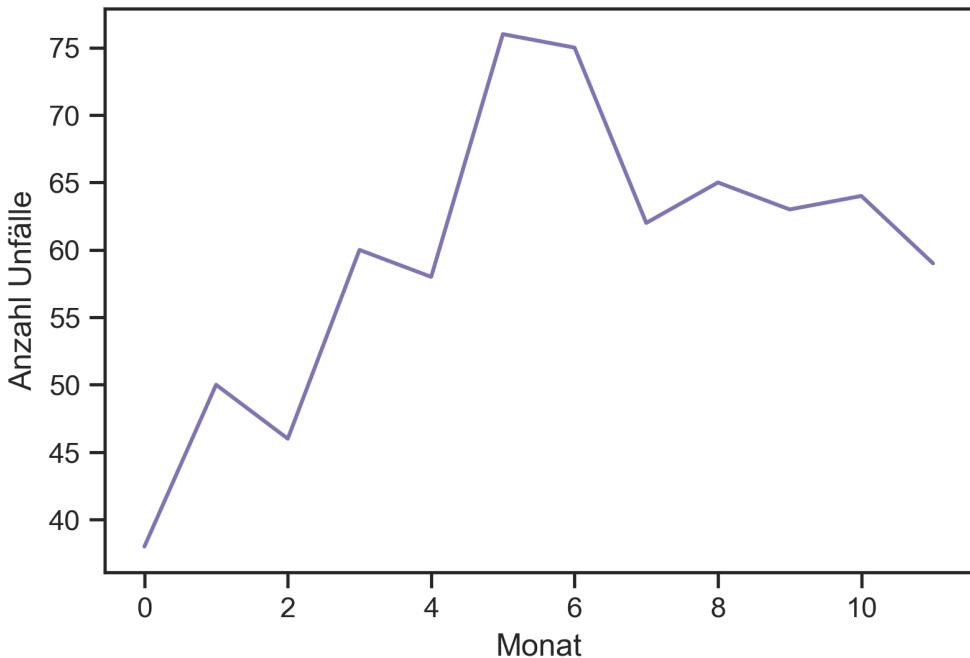


Abbildung 10: Unfälle in Mainz im Monat

Mit der Fertigstellung des Plots konnte mit der Programmierung der polynomiellen Regression begonnen werden. Hierfür musste zunächst der bestmögliche Grad der Funktion bestimmt werden. Wie in den theoretischen Grundlagen beschrieben, gibt der Grad die Anzahl der Steigungsänderungen an. Das Regressionsmodell beinhaltet den Accuracy Score, mit dem man feststellen kann, wie gut sich die Funktion auf die Daten anpasst. Somit wurden das Modell auf jeden Grad getestet, bis sich keine Verbesserung mehr des Scores ergeben hat. Dies trat bei dem Grad = 5, mit einem Score von 80,637% ein. Abschließend wurde der Wert, welcher die Anzahl der Monate für die Vorhersage angibt, als Input Variable definiert. Hierdurch kann der Nutzer des Programms direkt über die Konsole den Zeitraum für die Vorhersage eingeben.

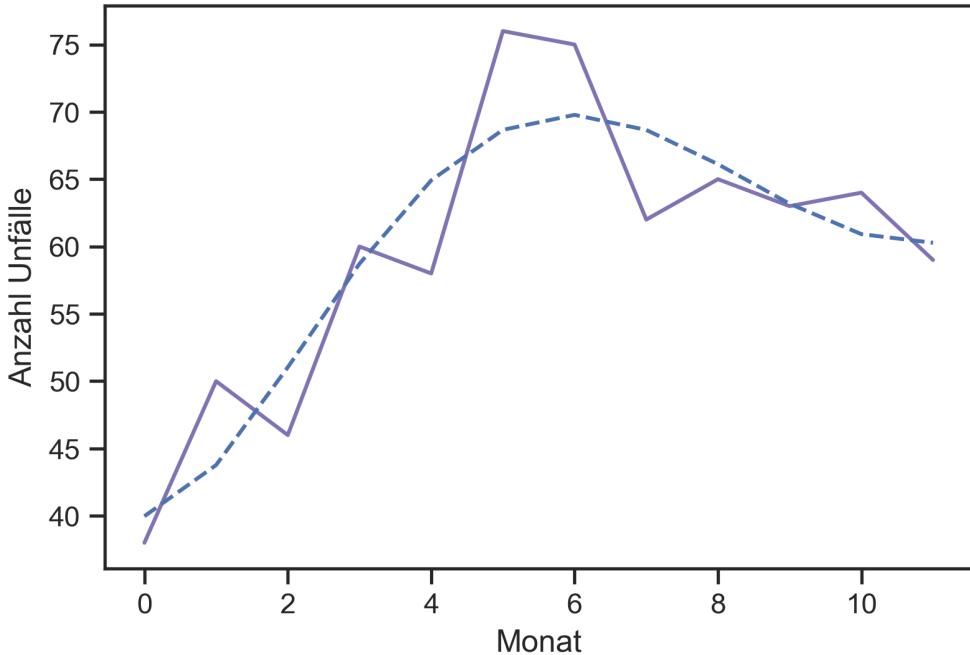


Abbildung 11: Regression für Mainz auf Monatsbasis

In der Abbildung 11 visualisiert die blau-gestrichelte Kurve die polynomielle Regression, welche sich auf die lila Kurve, also die Unfallzahlen, anpasst. Die Kurve passt sich nicht genau auf den Verlauf der Unfallzahlen an, wodurch der Overfitting-Effekt vermieden wird.

Bei einer Vorhersage nach einer Einheit, also einem Monat, sagt das Modell 61 Unfälle vorher. Der Durchschnittswert der 12 Unfallzahlen beträgt 60, weswegen eine Vorhersage von 61 valide erscheint. Das Modell funktioniert somit gut für die Vorhersage eines kurzen Zeitraums. Für längere Zeiträume scheint es jedoch ungeeignet zu sein. Bei einem Vorhersagezeitraum von 9 Monaten fängt das Modell an, negative Unfallzahlen vorherzusagen. Im Fall von 9 Monaten wurden -22 Unfälle vorhergesagt. Dies ist unmöglich, da Unfälle nur in einem natürlichen Zahlenbereich angegeben werden können.

4.1.2 Erweitertes Modell der polynomiellen Regression

Die polynomielle Regression scheint, zumindest für einen kurzen Vorhersagezeitraum, gut auf den Daten zu funktionieren. Aus diesem Grund kann das Modell im nächsten Schritt auf einen Datensatz mit täglichen Unfallzahlen getestet werden, um hierdurch eine tägliche Vorhersage für die Anzahl der Unfälle mit Personenschaden zu generieren.

Als einzige Maßnahme hierfür musste lediglich der Dataframe umstrukturiert werden. Das Modell selbst bedarf keiner Veränderung. Zunächst wurde der Dataframe wieder auf die Informationen aus der Stadt Mainz gefiltert. Außerdem wurde wieder eine Zeitreihe erstellt. Im Gegensatz zur vorherigen Zeitreihe werden dieses Mal die Tage im Jahr 2016 und die jeweilige Anzahl an Unfällen an den Tagen ausgegeben. Dabei werden aufgrund des Datensatzes nicht alle 365 Tage im Jahr angegeben, sondern jeweils die Wochentage eines Monats.

Monat	1	2	3	...	83
Anzahl Unfälle	3	9	3	...	6

Tabelle 2: Dataframe für die täglichen Unfälle in Mainz

Auch dieser Dataframe wurde wieder in einem Plot veranschaulicht.

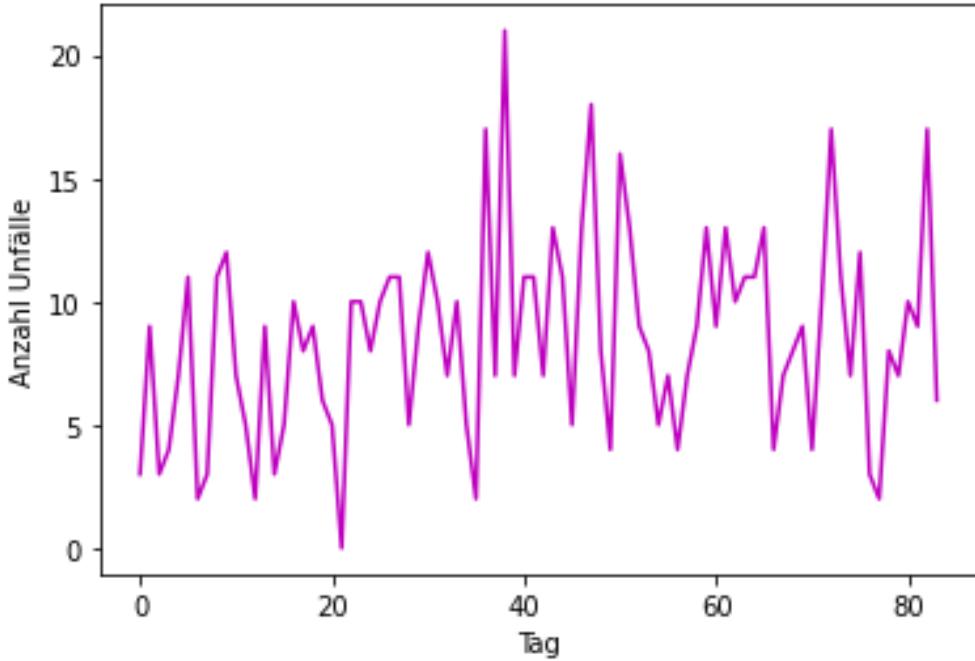


Abbildung 12: Unfälle in Mainz am Tag

Wie im vorherigen Modell musste im nächsten Schritt der optimale Grad für die polynomische Regression bestimmt werden. Auch im täglichen Vorhersagemodell wird erneut der Score der Regression als Grundlage für die Bewertung des Grades genommen. Bei dem Grad = 6 hat sich der Accuracy Score von 28,938% nicht weiter verbessert. Im Gegensatz zum vorherigen Modell ist der Score deutlich schlechter. Dies spiegelt sich auch in der Anpassung der Regressionskurve an die täglichen Unfälle wider.

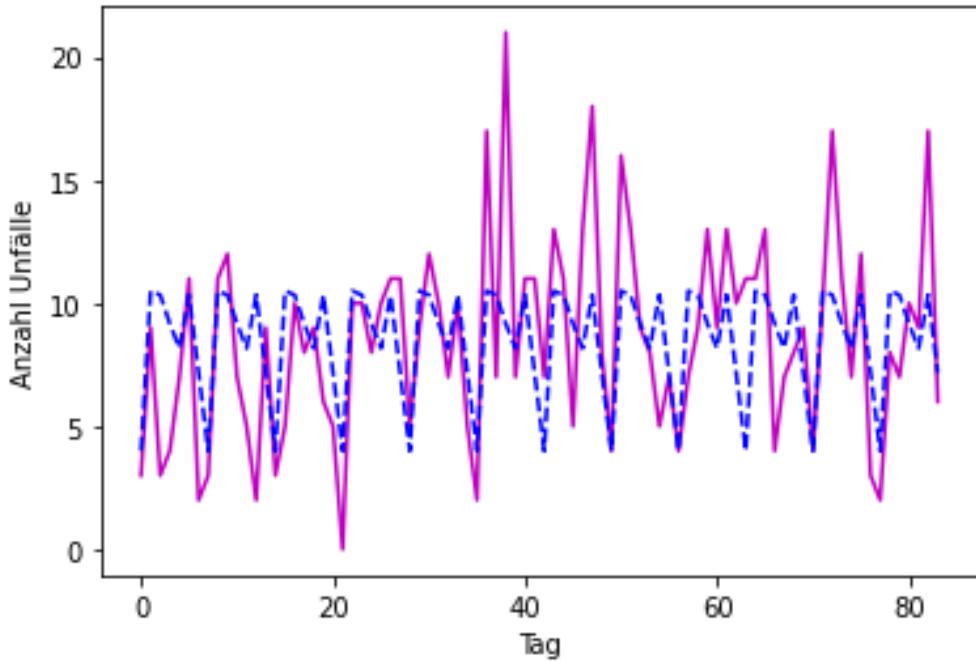


Abbildung 13: Regression für Mainz auf Tagesbasis

In der Abbildung 13 gibt die blau-gestrichelte Kurve wieder die polynomiale Regression an. Es ist zu erkennen, dass sich die Kurve nicht sonderlich auf die lila Kurve anpasst. Vielmehr lässt sich beim Verlauf der Regression eine Periodizität feststellen. Das bedeutet, der Verlauf der Kurve weist ein regelmäßiges oder auch zyklisches Verhalten auf. Die Regression scheint die lila Kurve zu ignorieren.

Durch die Accuracy und die schlechte Regression war abzusehen, dass die Vorhersage nicht gut werden würde. Bei einer Vorhersage nach einer Einheit, also einem Tag, sagt das Modell $-7\ 806\ 779\ 765$ Unfälle vorher. Abgesehen davon, dass die Zahl negativ ist, was bei der Anzahl der Unfälle nicht möglich sein kann, ist die Vorhersage absolut unrealistisch. Dies lässt sich verdeutlichen, indem die Vorhersage geplottet wird.

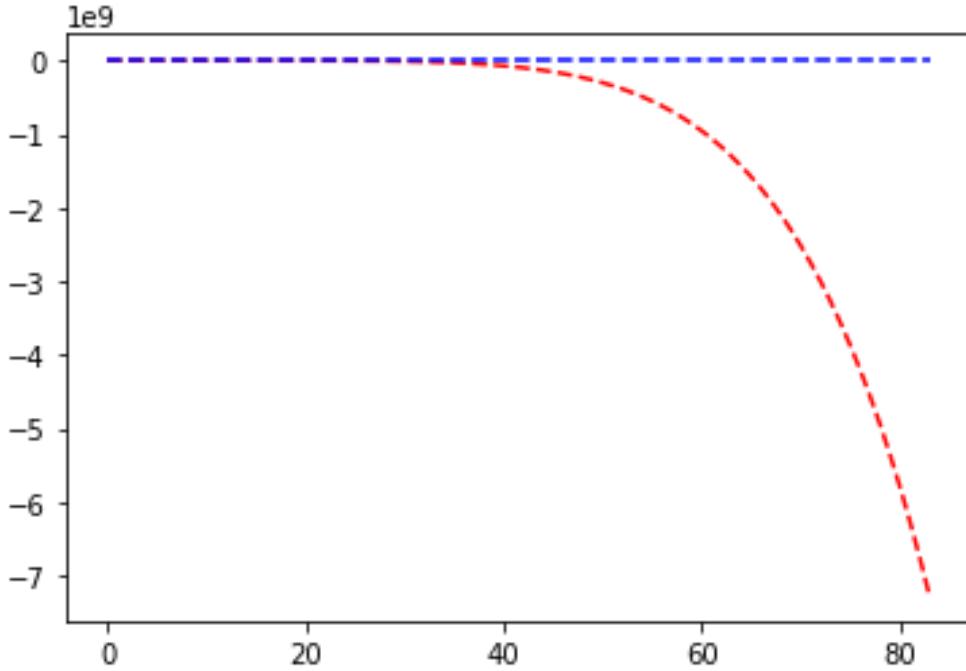


Abbildung 14: Vorhersage für Mainz auf Tagesbasis

Die blau-gestrichelte Kurve stellt wieder die Regression dar. Da die Vorhersage ein so hoher Ausreißer ist, ist die Regressionskurve in der Abbildung 14 nicht wiederzuerkennen. Die rot-gestrichelte Kurve zeigt den Vorhersagezeitraum. Die rot-gestrichelte Kurve folgt dem Verlauf der blau-gestrichelten Kurve und bricht am Ende aus. Dieser Bereich ist der Vorhersagezeitraum. Die Vorhersagekurve weist einen extrem fallenden Verlauf auf, wodurch der unrealistische Vorhersagewert entsteht.

Im Gegensatz zum Vorgänger versagt das tägliche Vorhersagemodell bereits bei einem gerin- gen Vorhersagezeitraum. Der Grund hierfür ist höchstwahrscheinlich die ignorante Anpas- sungen der Regression. Nach längerem Forschen konnten wir die Ursache für dieses Phänomen herausfinden. Wie bereits beschrieben, wurde das Modell selbst nicht verändert. Lediglich der Input wurde angepasst, weswegen die Wahrscheinlichkeit hoch war, dass dies der Grund für die Periodizität war.

Wie bereits in der Beschreibung des Datensatzes erläutert, war es schwierig, die einzelnen Tage mit der jeweiligen Anzahl der Unfälle mit Personenschaden zu definieren. Der Grund hierfür ist, dass im Datensatz kein genaues Unfalldatum genannt wird, sondern lediglich der Wochentag, an dem ein Unfall stattgefunden hat. Deswegen haben wir die Tage im Datensatz so definiert, dass jeder Monat von Montag bis Sonntag geht und somit nur 7 Tage

beinhaltet. Da hierdurch ein periodischer 7-Tage-Rhythmus entstanden ist, konnte sich die Regressionskurve nicht an den Verlauf der Unfallkurve anpassen.

Die 'Reparatur' des Modells war relativ einfach. Der 7-Tage-Rhythmus musste lediglich in einen Index umgewandelt werden, welcher die Anzahl der Datenpunkte zählt. Des Weiteren wurde bei diesem Mal die gesamten Informationen zu den Unfällen in Mainz über 5 Jahre miteinbezogen. Dieser recht simple Eingriff hat die Leistung des Modells erheblich verändert, was im Plot 15 zu erkennen ist.

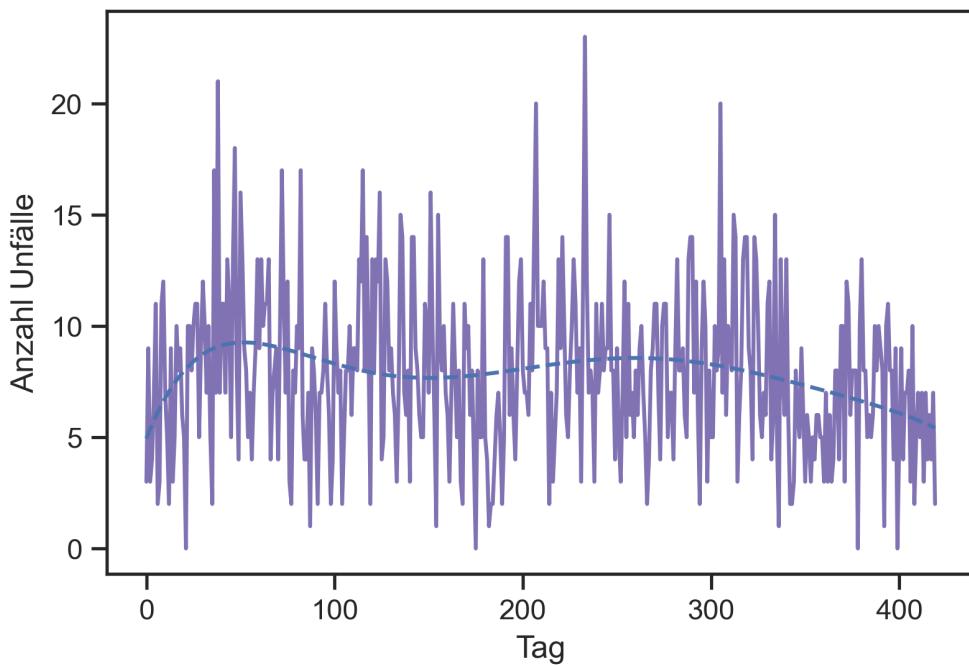


Abbildung 15: Neue Regression für Mainz auf Tagesbasis

Die Abbildung 15 zeigt auf der lila Kurve den Verlauf der Unfälle mit Personenschaden in Mainz von den Jahren 2016 bis 2020. Die blau-gestrichelte Kurve zeigt die polynomische Regression. Im Vergleich zum fehlerhaften Modell, passt sich die Regression nun deutlich besser an die Zeitreihe der Unfallzahlen an. Trotz dieser Verbesserung ist der Score für die Anpassung nach wie vor schlecht. Der beste Accuracy Score liegt bei 5,6%, bei einem Grad von 6. Somit scheint das Modell noch immer mit der hohen Menge an Daten überfordert zu sein.

Um diese These zu bestätigen, wurden drei F1 Scores als weiteres Bewertungsmaß für das Modell herangezogen. Der 'macro' f1-Score berechnet für jedes Label den ungewichteten Mittelwert und liegt bei 0,086%. Der 'micro' f1-Score berechnet die Summe aus echt positiven, falsch negativen und falsch positiven Werten und liegt bei 0,952%. Der 'weighted' f1-Score

berechnet für jedes Label den Mittelwert, gewichtet nach der Anzahl der wahren Instanzen und liegt bei 0,018%. Die Ergebnisse aus der Analyse der F1 Scores bestätigt die These. Da alle Werte schlecht sind, scheint das Modell nach wie vor ungeeignet für die Daten zu sein.

Tag	1	...	9	...	25	...	36	...	45	...	52	...	63
Anzahl Unfälle	5	5	4	4	3	3	2	2	1	1	0	0	-1

Tabelle 3: Vorhersagen der polynomiellen Regression

Die Tabelle 3 zeigt die vorhersagten Unfallzahlen für einen Zeitraum von 1 bis 63 Tagen. Es lässt sich ein Vorhersagetrend durch die polynomielle Regression feststellen. Die Regression beginnt damit 5 Unfälle an einem Tag vorherzusagen und geht dann schrittweise je um einen Unfall runter, bis sie schlussendlich im negativen Bereich landet. Ab einer Vorhersage von 63 Tagen wird das Modell somit unrealistisch, da es keine Unfallzahlen im negativen Zahlenbereich geben kann.

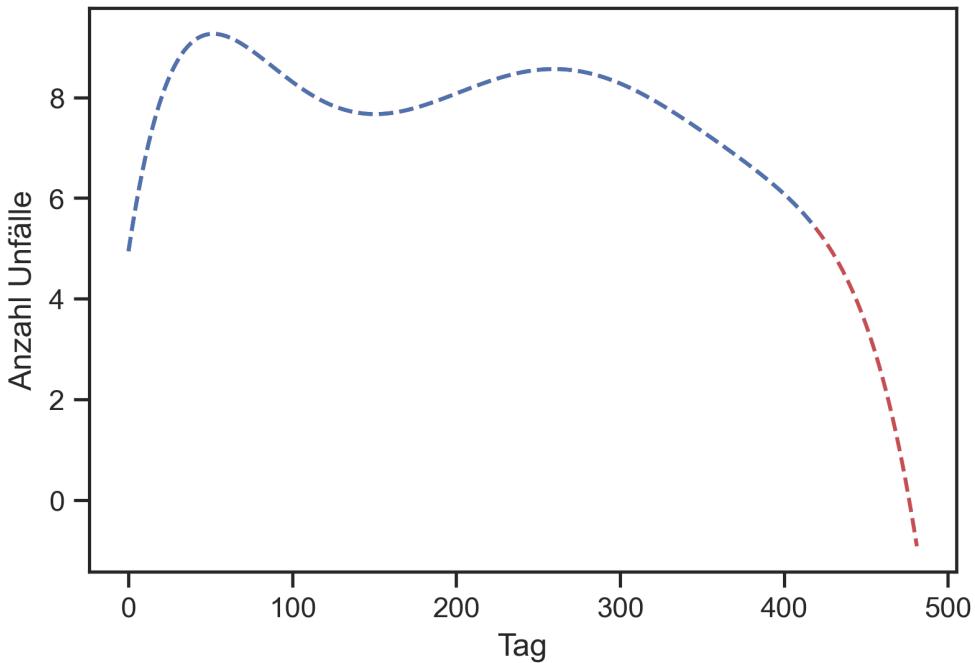


Abbildung 16: Neue Vorhersage für Mainz auf Tagesbasis

In der Abbildung 16 ist der Vorhersagetrend nun viel ersichtlicher, als im fehlerhaften Modell. Die rot-gestrichelte Vorhersagekurve unterstreicht die Erkenntnisse aus der Tabelle 3. Die polynomielle Regression beginnt mit einer Vorhersage von 5 Tagen und steigt dann sukzessive um jeweils eine Einheit ab, weswegen sie zwangsläufig in den negativen Bereich gelangt.

4.2 Vorhersage mithilfe eines künstlichen neuronalen Netzes

Im Rahmen dieser Arbeit kommt ein künstliches neuronales Netz zum Einsatz, welches mit Hilfe der Python Bibliothek Keras im Zusammenhang mit der von Google entwickelten Tensorflow Umgebung implementiert wird.

4.2.1 Trainings-, Validierungs- und Testdaten

Damit ein neuronales Netz trainiert werden kann, benötigt es Trainings-, Validierungs und Testdaten. Die hier genutzten Daten werden in Kapitel 3.1 und 3.2 genauer erläutert. Hier sei nur erwähnt, dass die erfassten Daten der „statistischen Ämter des Bundes und der Länder“ zu Straßenverkehrsunfällen mit Personenschaden als Hauptdatensatz dient. Zum Vorhersagen der Unfallanzahl zu einem Zeitpunkt wurde der Datensatz jedoch auf die Zeitpunkte der Jahre und der entsprechenden Unfallanzahl zu den Zeitpunkten reduziert. Zusätzlich werden Datensätze des deutschen Wetterdienstes zu den Trainingsdaten hinzugezogen, um eine Aussage darüber treffen zu können, ob dies die Vorhersage verbessert. Die hier genutzten Datensätze des Deutschen Wetterdienstes beinhalten die stündlichen Niederschlagswerte und die stündlich erfassten Lufttemperaturwerte der hier betrachteten Städte Mainz und Wiesbaden.

Insgesamt stehen für jede Stadt jeweils 10080 Daten zur Verfügung, die wiederum in Trainings-, Validierung- und Testdaten aufgeteilt werden. Die Anzahl der Daten kommt folgendermaßen zusammen:

$$\# \text{Jahre} \cdot \# \text{Monate} \cdot \# \text{Wochentage} \cdot \# \text{Stunden} = 5 \cdot 12 \cdot 7 \cdot 24 = 10080$$

Die 10080 Daten werden dabei wie folgt in Trainings-, Validierung- und Testdaten aufgeteilt:

	Insgesamt	Trainingsdaten	Validierungsdaten	Testdaten
# Instanzen	10080	7257	806	2016
rel. Anteil	1	0.72	0.08	0.2

Tabelle 4: Aufteilung der vorliegenden Daten in Trainings-, Validierungs- und Testdaten.

Um im späteren Verlauf ein geeignetes Bewertungsmaß für die Voraussage des künstlichen neuronalen Netzes zu wählen, zeigen die Tabellen 5 und 6 die Aufteilung der Daten in die vorhandenen Klassen (Label) auf.

# Unfälle	0	1	2	3	4	5
# Instanzen	7528	1936	495	98	22	1
rel. Häufigkeit	0.75	0.19	0.05	0.001	0.002	0.0001

Tabelle 5: Klassenaufteilung der Unfallanzahlen zu einem Zeitpunkt in der Stadt Mainz.

# Unfälle	0	1	2	3	4	5	6
# Instanzen	6413	2483	879	222	61	18	4
rel. Häufigkeit	0.63	0.25	0.09	0.022	0.006	0.002	0.0004

Tabelle 6: Klassenaufteilung der Unfallanzahlen zu einem Zeitpunkt in der Stadt Wiesbaden.

Hierbei wird deutlich, dass es sich um sehr unbalancierte Daten handelt, da ca. 65 – 75% (je nach Stadt) der Daten in einer Klasse befinden. Daher wird für den weiteren Verlauf dieser Arbeit vor allem der f1-Score zur Bewertung des künstlichen neuronalen Netzes genutzt. Die Berechnung des f1-Scores ist in Kapitel 2.1.1 beschrieben.

4.2.2 Aufbau des künstlichen neuronalen Netzes

Im Folgenden wird der Aufbau des in dieser Arbeit implementierten künstlichen neuronalen Netzes erläutert. Ein künstliches neuronales Netz besteht, simpel formuliert, aus einem Input-Layer, einem oder mehreren Hidden-Layern und einem Output-Layer [1, 7, 16].

Der Input-Layer des hier genutzten künstlichen neuronalen Netzes besteht jeweils aus vier oder sechs künstlichen Neuronen. Dies entspricht der Anzahl an Attributen, die genutzt werden. Beim Trainieren des künstlichen neuronalen Netzes mithilfe der Zeitpunkte werden vier künstliche Neuronen benötigt (Jahr, Monat, Tag, Stunde) und im Falle des Hinzuziehens der Wetterdaten zwei weitere künstliche Neuronen für die Temperaturwerte und die Niederschlagswerte zu den entsprechenden Stunden. In dieser Arbeit soll die Unfallanzahl zu einem Zeitpunkt vorhergesagt werden. Somit ergeben sich als Labels des künstlichen neuronalen Netzes die möglichen Unfallanzahlen an. So gibt z. B. das Label 0 die Klasse an, in der keine Unfälle passiert sind. Die Anzahl der künstlichen Neuronen im Outputlayer entspricht hier der Anzahl der Labels. Im Falle des Trainings auf dem Datensatz der Stadt Mainz sind es sechs künstliche Neuronen und im Falle der Stadt Wiesbaden sieben künstliche Neuronen (siehe Tabellen 5 und 6). Die Anzahl der Output-Neuronen weicht voneinander ab, da in der Stadt Wiesbaden zu einem Zeitpunkt maximale sechs Unfälle mit Personenschäden registriert wurden und in der Stadt Mainz maximal fünf zu einem Zeitpunkt.

Um die ideale Anzahl an künstlichen Neuronen in dem Hidden-Layer zu ermitteln, werden künstliche neuronale Netze mit 10–100 (10er-Schritte) künstlichen Neuronen im Hidden-Layer trainiert. Abbildung 17 zeigt die f1-Scores in Abhängigkeit der Anzahl der künstlichen Neuronen für zwei Durchläufe. Im Laufe dieser Arbeit wurden die künstlichen neuronalen Netze wesentlich häufiger als hier dargestellt trainiert und bewertet.

Abbildung 17 zeigt, dass wenig Zusammenhang zwischen der Anzahl der künstlichen Neuronen im Hidden-Layer und dem f1-Score zu erkennen ist. Grundsätzlich lässt sich in Abbildung 17 auch erkennen, dass der f1-Score nicht besonders hoch liegt und somit keine guten Ergebnisse erzielt werden. Somit wird die Anzahl der künstlichen Neuronen im Hidden-Layer frei gewählt und auf 50 gesetzt. Abbildung 18 zeigt den Aufbau der hier genutzten künstlichen neuronalen Netze.

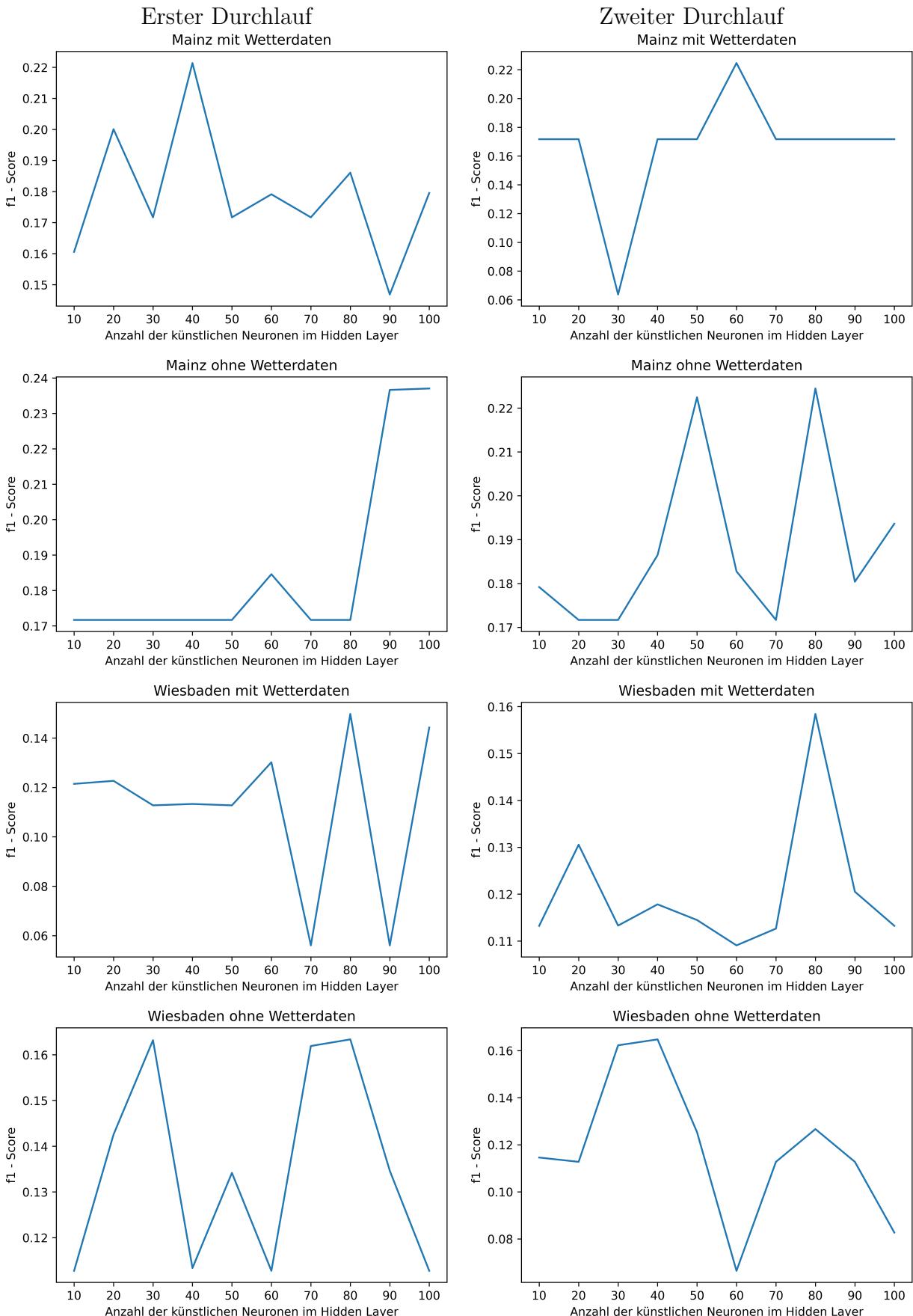


Abbildung 17: Entwicklung des f1-Scores bei ansteigender Anzahl der künstlichen Neuronen im Hidden-Layer. Links zeigt den ersten Durchlauf für jeweils beide Städte, einmal mit Einbeziehen der Wetterdaten und einmal ohne. Die Plots rechts zeigen den zweiten Durchlauf.

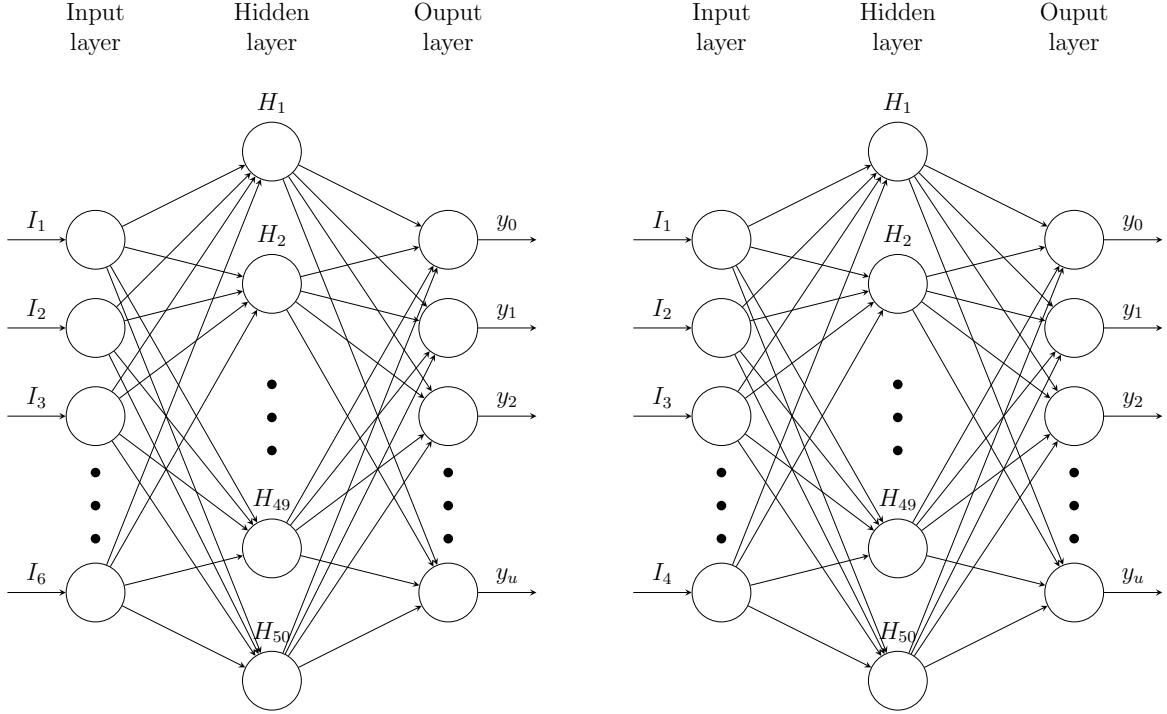


Abbildung 18: Aufbau der künstlichen neuronalen Netze zur Vorhersage der Unfallanzahl mit Personenschaden zu einem bestimmten Zeitpunkt. Links für das Training mit Einbezug der Wetterdaten als Input. Rechts für das Training ohne Einbezug der Wetterdaten als Input. u gibt die maximale Anzahl an Unfällen an, die zu einem Zeitpunkt in der Stadt registriert worden sind, die als Datensatz genutzt wird.

Weiterhin müssen für die einzelnen Layer Aktivierungsfunktionen gewählt werden. Die im Folgenden beschriebenen Einstellungen gelten für beide künstlichen neuronalen Netze. Für den Input-Layer wird keine Aktivierungsfunktion benötigt, da der Input die unveränderten Attribute sind und im Input-Layer keine Berechnung erfolgt. Für die Hidden-Layer wird die Aktivierungsfunktion „ReLU“ genutzt. Die ReLU-Funktion gibt ab einem Wert $x \geq 0$ den Wert x aus und ist somit einfach zu berechnen. Eine genaue Beschreibung der ReLU-Funktion und weiterer Aktivierungsfunktionen findet sich in [5, 13]

Da es mehr als nur ein Label gibt, liegt ein sogenanntes „Multiclass multilabel classification“ Problem vor. Aus diesem Grund wird für den Output-Layer die Aktivierungsfunktion „Softmax“ gewählt [7, 24]. Dadurch geben die Output-Neuronen einen Wert zwischen 0 und 1 aus, die aufsummiert genau 1 ergeben. Der Output-Wert der einzelnen künstlichen Neuronen entspricht dabei der Wahrscheinlichkeit, wie viele Unfälle zu einem Zeitpunkt verursacht werden.

betrachtetes Output-Neuron (Label)	0	1	2	3	4	5
Output des KNN	0.7532	0.2455	0.0001	0.0012	0	0
Gewünschter Output	1	0	0	0	0	0

Tabelle 7: Beispielhafte Vorhersage eines künstlichen neuronalen Netzes für die Stadt Mainz

Im Falle der beispielhaften Vorhersage eines künstlichen neuronalen Netzes für die Stadt Mainz in Tabelle 7 sagt das künstliche neuronale Netz voraus, dass mit einer Wahrscheinlichkeit von 75% kein Unfall verursacht wird, mit einer Wahrscheinlichkeit von 25% ein Unfall verursacht wird und mit annähernd 0% zwei oder mehr Unfälle verursacht werden.

In dieser Arbeit wird als Optimierer der „Adaptive Moment Estimation“ (Adam) Optimierer gewählt. Bei diesem Optimiere werden im Gegensatz zu anderen Optimieren die aussagekräftigsten Voraussagen getroffen. Eine genaue Beschreibung des Optimierers adam und weitere Optimierer finden sich in den Referenzen [4, 11].

Die Anzahl der trainierten Epochen wurde gleich 200 gewählt. Ebenso wie bei der Anzahl der künstlichen Neuronen im Hidden-Layer, wird auch hier kein Zusammenhang erkennbar, dass zum Beispiel bei einer höheren Anzahl an Epochen der f1-Score verbessert wird. Somit wurde die Anzahl der Epochen ebenfalls frei gewählt.

4.2.3 Ergebnisse des künstlichen neuronalen Netzes

Die implementierten künstlichen neuronalen Netze wurden im Laufe dieser Arbeit, um ein Vielfaches von dem, was hier dargestellt ist, trainiert und untereinander verglichen. Da wie in Kapitel 18 beschrieben kein Zusammenhang zwischen den Aufbauten der künstlichen neuronalen Netze und deren Ergebnisse zu erkennen ist, werden die Einstellungen der Hyperparameter (Anzahl der Hiddenlayer, Anzahl der Neuronen, Einstellung des Optimierers) frei gewählt.

Es wurden insgesamt vier unterschiedliche Datensätze zum Trainieren genutzt.

1. Unfalldaten der Stadt Mainz ohne Wetterdaten
2. Unfalldaten der Stadt Mainz mit Wetterdaten
3. Unfalldaten der Stadt Wiesbaden ohne Wetterdaten
4. Unfalldaten der Stadt Wiesbaden mit Wetterdaten

Testen der künstlichen neuronalen Netze auf den jeweiligen Testdatensätzen

In Tabelle 8 sind die erzielten f1-Scores der einzelnen künstlichen neuronalen Netze bezüglich der jeweiligen Testdatensätze zusammengefasst. Die genutzten künstlichen neuronalen Netze sind dabei jeweils mit dem Trainingsdataset der gleichen Stadt trainiert worden. Alle künstlichen neuronalen Netze wurden mehrfach trainiert und getestet. Tabelle 8 zeigt dabei die Ergebnisse von zwei Durchläufen.

Stadt	Wetterdaten	f1-Score	
		1.Durchlauf	2.Durchlauf
Mainz	✓	0.1196	0.1717
Mainz	✗	0.1717	0.1814
Wiesbaden	✓	0.1503	0.1127
Wiesbaden	✗	0.1127	0.1127

Tabelle 8: F1-Scores der Vorhersage der Testdaten. Hierbei wird die Unfallanzahl der Testdaten mithilfe eines künstlichen neuronalen Netzes, welches mit den Trainingsdaten der jeweiligen Stadt trainiert wurde, vorhergesagt.

Die Werte zeigen, dass die Ergebnisse grundsätzlich nicht besonders gut sind (ein f1-Score von 1 ist gewünscht). Durch das Hinzuziehen eines weiteren Datensatzes, den Wetterdaten der jeweiligen Städte, ist keine klare Verbesserung der Vorhersagen zu erkennen. Für das Testen des künstlichen neuronalen Netzes der Stadt Mainz wurde sogar in beiden Durchläufen ohne Einbeziehen der Wetterdaten ein besserer f1-Score erzielt.

Hier wurde jedoch nur der f1-Score im Gesamten betrachtet. Es kann auch interessant sein, sich den f1-Score jeder einzelnen Klasse anzuschauen, um beurteilen zu können, wie gut oder schlecht das künstliche neuronale Netz auf den jeweiligen Klassen performt.

Die Abbildungen 19 und 20 stellen die Ergebnisse des Trainings in Form der f1-Scores der einzelnen Klassen dar. Abbildung 19 stellt dabei die f1-Scores im ersten Durchlauf für jede Klasse dar und Abbildung 20 die f1-Scores für den zweiten Durchlauf.

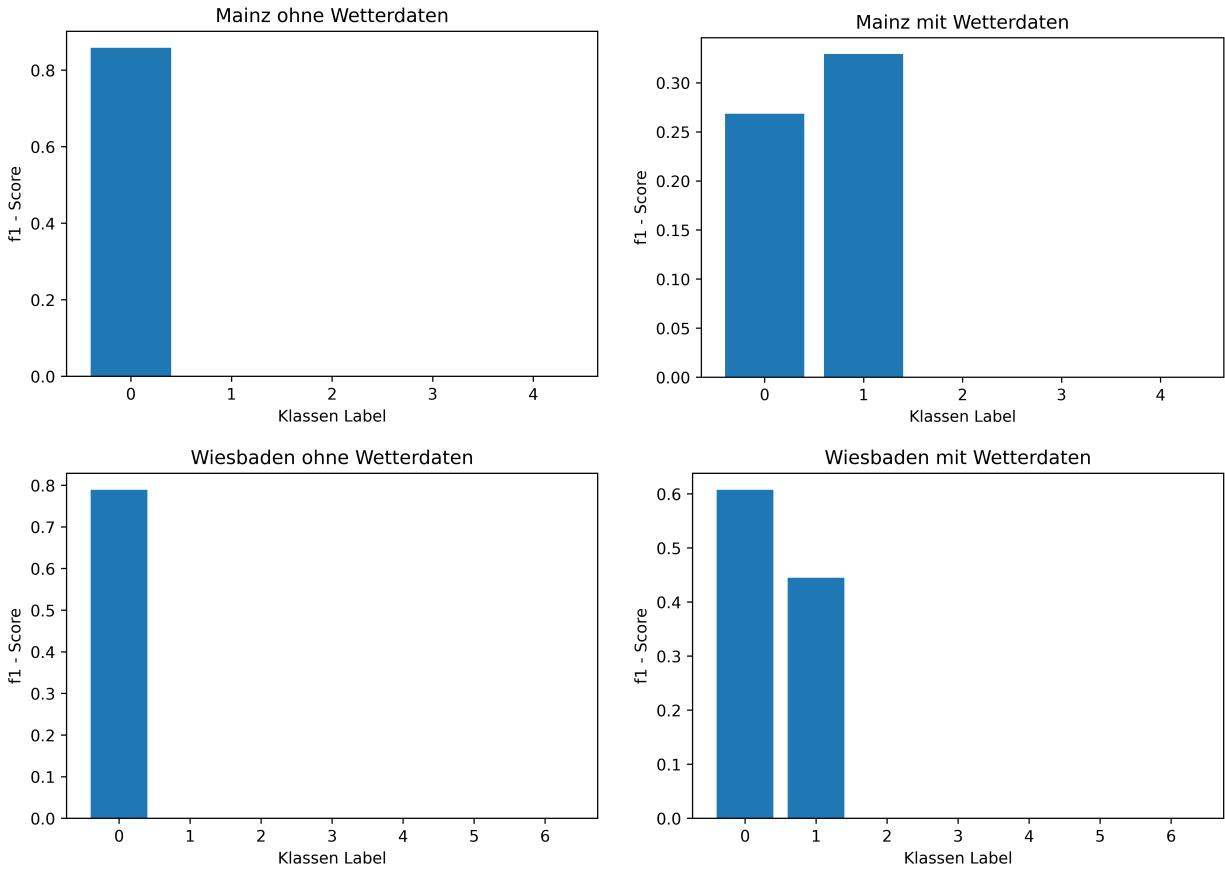


Abbildung 19: Erster Durchlauf: F1-Score jeder Klasse bei Vorhersage bezüglich der Testdaten, nach dem Training mit den Daten der Stadt Mainz (oben) bzw. der Stadt Wiesbaden (unten). Links ohne, rechts mit Einbezug der Wetterdaten.

In den Abbildungen 19 und 20 ist deutlich zu sehen, dass so gut wie jedes trainierte künstliche neuronale Netz in der Klasse 0 gut performt (f1-Score von ca. 0.8) dafür aber in den anderen Klassen meist einen f1-Score von 0 erzielt. Dies spiegelt gut wider, dass es sich bei dem hier genutzten Datenset um ein sehr unbalanciertes handelt (siehe Kapitel 4.2.1).

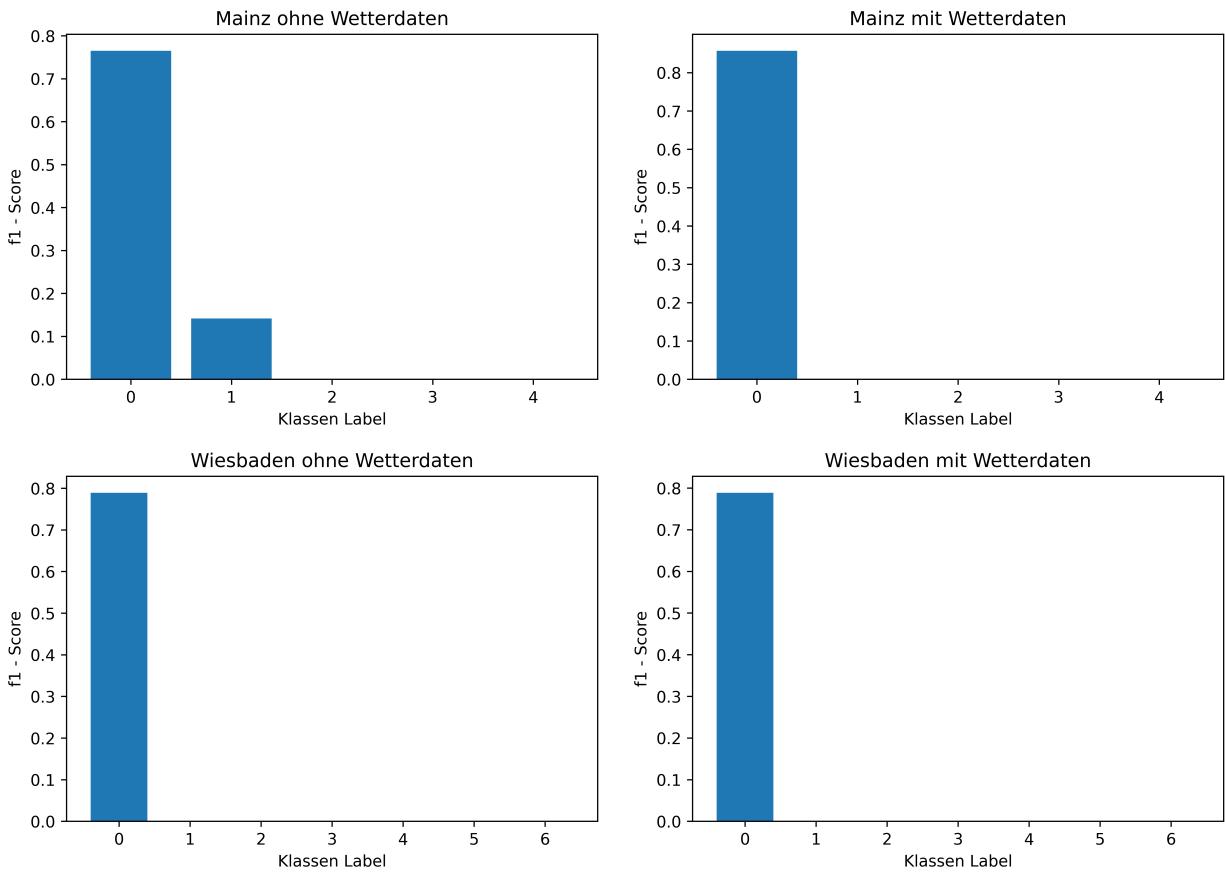


Abbildung 20: Zweiter Durchlauf: F1-Score jeder Klasse bei Vorhersage bezüglich der Testdaten nach dem Training mit den Daten der Stadt Mainz (oben) bzw. der Stadt Wiesbaden (unten). Links ohne, rechts mit Einbezug der Wetterdaten.

Testen der künstlichen neuronalen Netze auf den Datensatz einer anderen Stadt

Im Folgenden sollen die Unfallanzahlen zu den jeweiligen stündlichen Zeitpunkten der Jahre 2016 bis 2020 der Städte Mainz und Wiesbaden vorausgesagt werden. Zur Vorhersage werden jeweils die auf der anderen Stadt trainierten künstlichen neuronalen Netze genutzt. So wird das künstliche neuronale Netz, welches auf den Daten der Stadt Mainz trainiert wurde, zur Vorhersage der Unfallanzahlen der Stadt Wiesbaden genutzt und umgekehrt.

Abbildung 21 zeigt hierbei alle vier Fälle grafisch auf. Hier werden die Differenzen der vorhergesagten Unfallanzahlen zu den tatsächlichen Unfallanzahlen dargestellt. Es ist zu erkennen, dass für die meisten Instanzen die richtige Anzahl an Unfällen vorausgesagt wird und die Differenz zur tatsächlichen Unfallanzahl somit 0 beträgt. In wenigen Fällen werden weniger oder mehr vorausgesagt.

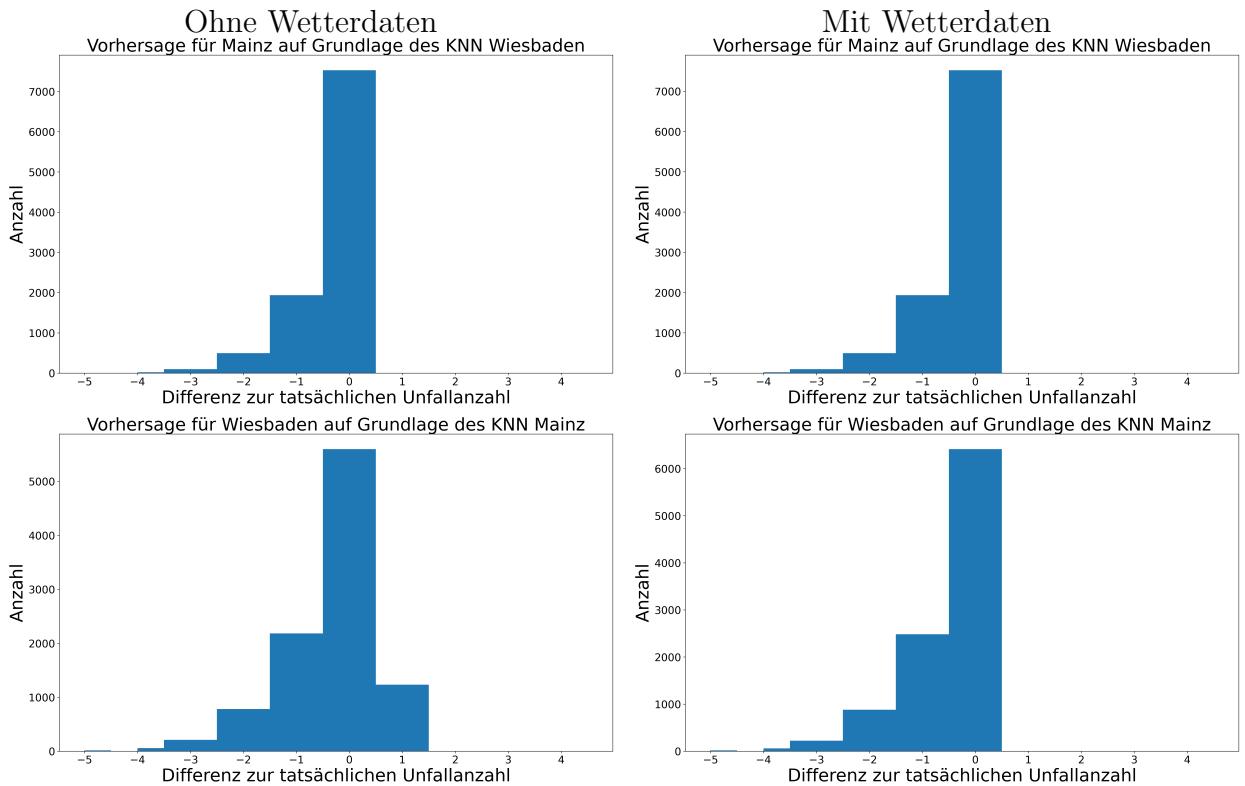


Abbildung 21: Differenzen der vorhergesagten Unfallanzahlen zu den tatsächlichen Unfallanzahlen. Die Daten, für die die Vorhersage getätigt wird, sind alle vorhandenen Daten der jeweiligen Stadt (2016-2020). Die Vorhersagen werden mithilfe des künstlichen neuronalen Netzes getroffen, welches mit den Daten der jeweils anderen Stadt trainiert wurde

Ein Blick auf die f1-Scores in Tabelle 9 zeigt jedoch auf, dass auch hier keine besonders gute Voraussage getroffen wird. Der Anschein kommt alleine daher, dass ca. 65 bis 75% (je nach betrachteter Stadt) der Daten der Klasse 0 zugeordnet sind, sodass das künstliche neuronale Netz schlecht trainiert und immer vorhersagt, dass 0 Unfälle mit Personenschäden verursacht werden. Aus genau diesem Grund ist es bei unbalancierten Datensätzen sinnvoll als Bewertungsmaß nicht nur auf den Accuracywert zurückzugreifen.

Stadt	Wetterdaten	f1 - Score	accuracy
Mainz	✓	0.14	0.75
Mainz	✗	0.14	0.75
Wiesbaden	✓	0.11	0.64
Wiesbaden	✗	0.13	0.56

Tabelle 9: F1-Score und Accuracy der Vorhersage der Unfallanzahl einer Stadt mithilfe eines künstlichen neuronalen Netzes, welches mit den Trainingsdaten der jeweiligen anderen Stadt trainiert wurde.

5 Zusammenfassung

Das Ziel dieser Arbeit war es, die Unfallanzahl mit Personenschaden in einer Stadt vorherzusagen. Dazu wurden die Vorhersagemethoden der polynomiellen Regression und der künstlichen neuronalen Netze genutzt. Dabei wurde auch betrachtet, ob eine Vorhersage der beiden Methoden durch das Hinzufügen eines weiteren Datensatzes verbessert werden kann.

Zum Einstieg wurden die Begriffe rund um die künstliche Intelligenz erläutert. Dazu gehört der Begriff des Machine Learning. Das Machine Learning nimmt in unserer Welt einen immer größeren Stellenwert ein, da es dadurch möglich ist, Probleme zu lösen, die vorher nicht oder nur schlecht lösbar waren. Dem Machine Learning werden die zwei Vorhersagemethoden der polynomiellen Regression und der künstlichen neuronalen Netze zugeordnet.

Die Vorhersage von Unfallzahlen mit Personenschaden mittels einer polynomielle Regression konnte erfolgreich implementiert und angewendet werden. Jedoch ist das Modell mit der Höhe der Datenmenge überfordert. Die Vorhersage spiegelt sich in einem Trend wider, welcher wenig valide Ergebnisse präsentiert. Somit lässt sich erschließen, dass eine polynomielle Regression für die Vorhersage von Unfallzahlen ungeeignet ist.

Es bestehen diverse Möglichkeiten, das hier vorgestellte Modell der polynomiellen Regression zu erweitern und eventuell zu verbessern. Beispielsweise könnten weitere Informationen, wie Daten zu Temperatur und Niederschlag, hinzugefügt werden. Hierdurch würde das Modell in eine multivariate polynomielle Regression umgewandelt werden. Da das Modell jedoch bereits in seiner univariaten Form keine guten Ergebnisse erzielt, ist es vermutlich nicht zielführend, weitere Forschungsarbeit in die polynomielle Regression zur Vorhersage von Unfallzahlen zu investieren.

Eine Vorhersage mittels künstlicher neuronaler Netze konnte nur in der Labelklasse 0 gute Ergebnisse erzielen, da dort auch die meisten Daten zu Verfügung standen. Insgesamt ist die Vorhersage der Unfallanzahl jedoch sehr ungenau. Ein Ansatz um die Voraussage gegebenenfalls verbessern zu können ist es mehr Daten und genauere Daten zu den Unfällen zu sammeln. Das Problem des unbalancierten Datensatzes bleibt jedoch vermutlich weiterhin bestehen, da auf alle Zeitpunkte des Jahres betrachtet deutlich mehr Zeitpunkte keinen Unfall registrieren als umgekehrt. Es ist denkbar, dass künstliche neuronale Netze bei einer deutlich größeren Stadt, in der mehr Unfälle verursacht werden, bessere Vorhersagen erzielen können. Dadurch könnte der Datensatz etwas balancierter sein.

Literatur

- [1] 3BLUE1BROWN. But what is a neural network? <https://www.youtube.com/watch?v=aircArUvnKk>, 2021. Accessed: 2021-14-07.
- [2] AMAZON. Wie unsere wissenschaftler alexa schlauer machen. <https://blog.aboutamazon.de/amazon-ger%C3%A4te/wie-unsere-wissenschaftler-alexa-schlauer-machen>, 2018. Accessed: 2022-23-02.
- [3] BRADLEY, R. A., AND SRIVASTAVA, S. S. Correlation in polynomial regression. *The American Statistician* 33, 1 (1979), 11–14.
- [4] BROWNLEE, J. Gentle introduction to the adam optimization algorithm for deep learning. <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>, 2017. Accessed: 2022-25-06.
- [5] BROWNLEE, J. A gentle introduction to the rectified linear unit (relu). <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>, 2019. Accessed: 2022-25-06.
- [6] BUCKLAND, M., AND GEY, F. The relationship between recall and precision. *Journal of the American society for information science* 45, 1 (1994), 12–19.
- [7] CHOLLET, F. *Deep learning with Python*. Simon and Schuster, 2017.
- [8] DEO, R. C. Machine learning in medicine. *Circulation* 132, 20 (2015), 1920–1930.
- [9] DEUTSCHER WETTERDIENST. Historische stündliche Stationsmessungen der Lufttemperatur und Luftfeuchte für Deutschland. https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/hourly/air_temperature/historical/, 2022. Accessed: 2022-26-05.
- [10] DEUTSCHER WETTERDIENST. Historische stündliche Stationsmessungen der Niederschlagshöhe für Deutschland. https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/hourly/precipitation/historical/, 2022. Accessed: 2022-26-05.
- [11] DOSHI, S. Various optimization algorithms for training neural network. <https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6>, 2019. Accessed: 2022-25-06.

- [12] EL NAQA, I., AND MURPHY, M. J. What is machine learning? In *machine learning in radiation oncology*. Springer, 2015, pp. 3–11.
- [13] KERAS. Layer activation functions. <https://keras.io/api/layers/activations/>, 2022. Accessed: 2022-25-06.
- [14] KORSTANJE, J. The f1 score. <https://towardsdatascience.com/the-f1-score-be8c2bbc38aa6>, 2021. Accessed: 2022-25-06.
- [15] MOHRI, M., ROSTAMIZADEH, A., AND TALWALKAR, A. *Foundations of machine learning*. MIT press, 2018.
- [16] NIELSEN, M. A. *Neural networks and deep learning*, vol. 25. Determination press San Francisco, CA, 2015.
- [17] OSTERTAGOVÁ, E. Modelling using polynomial regression. *Procedia Engineering* 48 (2012), 500–506.
- [18] PRETZ, C. Ansatz zur lösung des bin-packing-problems mit künstlichen neuronalen netzen. Bachelorarbeit, 2021.
- [19] PRETZ, C. Probably approximately correct. Seminararbeit, 2022.
- [20] READING, D. Typesetting neural network diagrams with tex. <https://medium.com/momenton/typesetting-neural-network-diagrams-with-tex-4920b6b9fc19>, 2021. Accessed: 2021-14-07.
- [21] RICH, E. *Artificial intelligence*. McGraw-Hill New York, 1983.
- [22] STATISTISCHE ÄMTER DES BUNDES UND DER LÄNDER. Unfallatlas. https://unfallatlas.statistikportal.de/_opendata2021.html, 2022. Accessed: 2022-26-05.
- [23] STRECKER, S. *Künstliche Neuronale Netze: Aufbau und Funktionsweise*. Universitätsbibliothek, 2004.
- [24] TENSOFLOW. tf.keras.activations.softmax. https://www.tensorflow.org/api_docs/python/tf/keras/activations/softmax/, 2021. Accessed: 2021-14-07.
- [25] TENSORFLOW. Module: tf.keras.activations. https://www.tensorflow.org/api_docs/python/tf/keras/activations, 2021. Accessed: 2021-14-07.

- [26] TENSORFLOW. Tensorflow grundklassifizierung: Bilder von kleidung klassifizieren.
<https://www.tensorflow.org/tutorials/keras/classification>, 2021. Accessed: 2021-14-07.