

# Klassifikation von portugisischen Schulnoten mit Bayes Netzen

**Dokumentation**

**Bachelor of Science**

des Studiengangs Informatik

an der Dualen Hochschule Baden-Württemberg Stuttgart

von

**Tom Freudenmann**

05.05.2023

**Bearbeitungszeitraum**  
**Matrikelnummer, Kurs**  
**Dozent**

20.03 - 05.05.2023  
6378195, INF20D  
Prof. Dr. Dirk Reichardt

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>II</b>
<b>Tabellenverzeichnis</b>	<b>III</b>
<b>1 Einleitung</b>	<b>1</b>
<b>2 Datenanalyse</b>	<b>2</b>
2.1 Verteilung der Zieleigenschaft . . . . .	2
2.2 Verteilungen anderer Eigenschaften . . . . .	3
2.3 Erste Korrelationsmatrix . . . . .	4
2.4 Zweite Korrelationsmatrix . . . . .	5
<b>3 Entwicklung und Evaluation</b>	<b>6</b>
3.1 Allgemeiner Ablauf . . . . .	6
3.2 Großes Netz . . . . .	7
3.3 Getunttes Netz . . . . .	9
3.4 Verbesserungsansätze . . . . .	11
3.5 Evaluation der Ansätze . . . . .	11
<b>4 Anwendung</b>	<b>16</b>
<b>5 Zusammenfassung und Ausblick</b>	<b>18</b>
<b>Anhang</b>	<b>A</b>
Literatur . . . . .	A

# Abbildungsverzeichnis

2.1	Verteilung der Schulnoten . . . . .	3
2.2	Verteilung des Alters . . . . .	4
2.3	Verteilung der Abwesenheit . . . . .	4
3.1	Großes Bayes Netz mit allen Knoten . . . . .	8
3.2	Getunttes Bayes Netz mit allen Knoten . . . . .	10
3.3	Konfusionsmatrizen des besten Ansatzes . . . . .	14
3.4	Konfusionsmatrizen des getuntten Netzes ohne Train-Test-Split . . . . .	14
3.5	Konfusionsmatrizen des großen Netzes . . . . .	15
4.1	Tkinter Anwendung . . . . .	16

# Tabellenverzeichnis

3.1	Ergebnisse der verschiedenen Ansätze . . . . .	12
-----	--	----

# 1 Einleitung

Die Leistung von Schülern in Portugal wurde untersucht [1]. Dazu werden demografische Daten verwendet, sowie Daten der Schule erhoben [2]. Es soll eine Anwendung basierend auf Bayes Netzen entwickelt werden, die die Schulnoten des letzten Trimesters (G3) bestmöglich klassifiziert. Dabei muss es möglich sein, dass einzelne Werte nicht gegeben sind [2].

Für die Umsetzung stehen Daten aus [1] zur Verfügung, die mithilfe eines Jupyter-Notebooks ausgewertet werden. Danach lassen sich verschiedene Bayes Netze auf Basis der vorverarbeiteten Daten entwickeln, implementieren und evaluieren, bevor die Anwendung implementiert wird.

Diese Dokumentation beschreibt dabei die Datenanalyse, mit ihren Erkenntnissen (Kapitel 2) und geht auf verschiedene Bayes Netze (Kapitel 3) ein und evaluiert ihre Leistung. Zum Schluss wird die Anwendung vorgestellt (Kapitel 4) und auf die Zusammenfassung der gesammelten Erkenntnisse, sowie dem Ausblick auf weitere Anpassungen (Kapitel 5) eingegangen.

## 2 Datenanalyse

Im Rahmen dieses Projekts wird mit einem Datensatz von 599 Einträgen mit 34 Eigenschaften gearbeitet. In der Datei `src/data-analysis.ipynb` werden die Daten zuerst geladen und auf ihre Vollständigkeit geprüft. Dabei wird schnell deutlich, dass es keine fehlenden Daten, aber dafür eine undefinierte Zeile gibt.

Importiert man den Datensatz mit der Python-Bibliothek *pandas*, dann wird diese Spalte als `Unnamed: 33` angezeigt. Sie enthält Werte zwischen null und eins und könnte somit als Gewichte für ein späteres Training verwendet werden. Sollte das keinen oder einen negativen Einfluss auf die Performance haben, dann lässt sich die Spalte auch entfernen.

In den nächsten Schritten werden die Spalten, die binäre Werte darstellen, auf 0 bzw. 1 transformiert, sodass später eine Korrelationsmatrix erstellt werden kann. Alle anderen kategorischen Werte, die nicht auf eine numerische Skala transformiert werden können, wie der Beruf der Eltern, werden in mehrere Spalten aufgeteilt. Dadurch entsteht für jede Kategorie eine extra Zeile, die entweder den Wert 0 oder 1 (Nein bzw. Ja) hat.

### 2.1 Verteilung der Zieleigenschaft

Ein Teil der Datenanalyse ist das Betrachten der Datenverteilung und das entfernen bzw. behandeln von Ausreißern. In Abbildung 2.1 wird die Verteilung der zu klassifizierenden letzten Trimester-Note (G3) dargestellt. Mit ihrer Hilfe lassen sich folgende Erkenntnisse über den Datensatz gewinnen:

1. Ein Großteil der Daten liegt zwischen 7 und 17
2. Fast ein Sechstel der Noten sind eine 11
3. Für die Werte 2,3,4,20 liegen keine Datenpunkte vor
4. Nach dem Boxplot sind die Werte von 0 und 1 Ausreißer

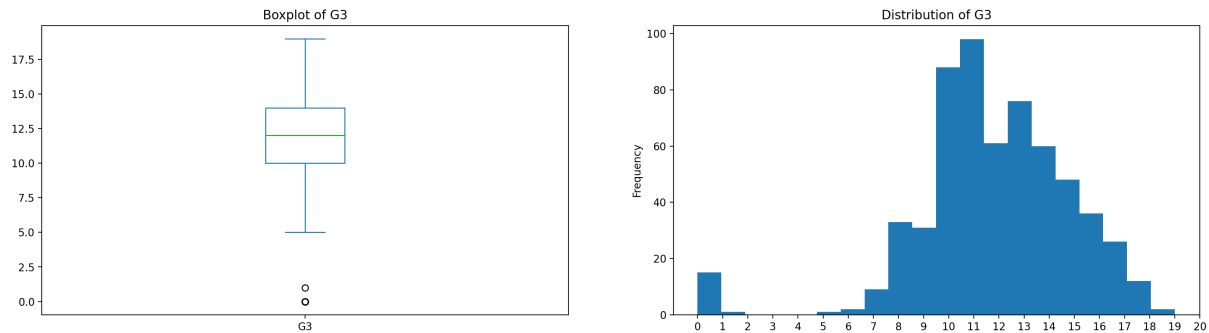


Abbildung 2.1: Verteilung der Schulnoten in einem Boxplot (links) und einem Balkendiagramm (rechts)

Untersucht man nun die Ausreißer genauer, dann fällt auf, dass die Noten gleich 0 wahrscheinlich von Schülern sind, die die Schule abgebrochen oder keine Noten bekommen haben. Diese Hypothese sollte man später für die Datenvorverarbeitung im Kopf behalten, weil das Entfernen von Ausreißern einen positiven Einfluss auf den Klassifikator haben kann.

Außerdem können fehlende oder wenig vertretene Werte zu einer schlechteren Performance des Netzes führen und es sollte in Betracht gezogen werden, die Werte kleiner 7 und größer 17 zusammenzufassen.

Zusätzlich zur G3-Verteilung werden im Notebook die Verteilungen von G1 und G2 betrachtet, die zu denselben Schlüssen von G3 führen und dadurch die Hypothese stützen.

## 2.2 Verteilungen anderer Eigenschaften

Im weiteren Verlauf des Notebooks werden die Verteilungen von Alter und Abwesenheiten genauer betrachtet, weil beide Spalten eine große Skala haben. Abbildung 2.2 zeigt die Verteilung des Alters und es fällt auf, dass die Mehrheit der Schüler zwischen 15 und 18 Jahre alt ist. Zusätzlich gibt es nur wenige, die älter als 20 sind, sodass das Alter ab 20 Jahren zusammengefasst werden kann.

Betrachtet man die in Abbildung 2.3 dargestellte Verteilung der Abwesenheiten, fällt auf, dass ab 15 Fehltagen kaum Daten vorliegen und der Boxplot die Datenpunkte als Ausreißer darstellt. Daraus folgt, dass auch hier der Wert auf 15 Fehltage begrenzt wird und alle Werte größer 15 auf einen Wert gemappt werden.

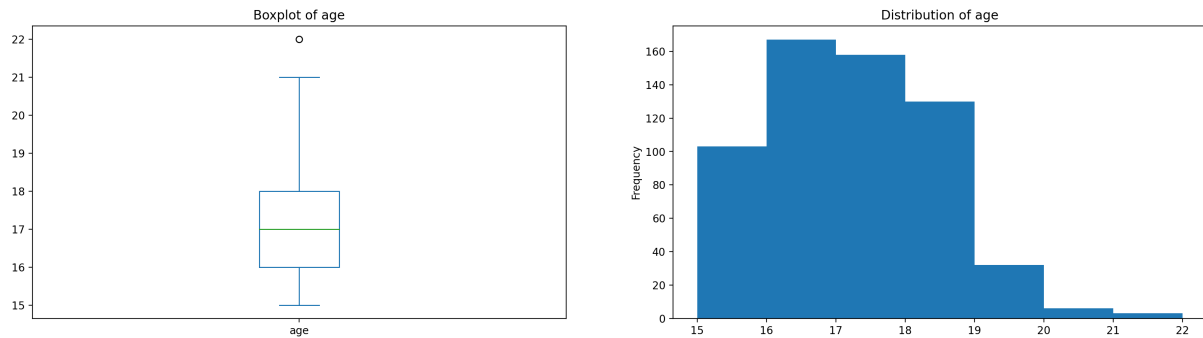


Abbildung 2.2: Verteilung des Alters in einem Boxplot (links) und einem Balkendiagramm (rechts)

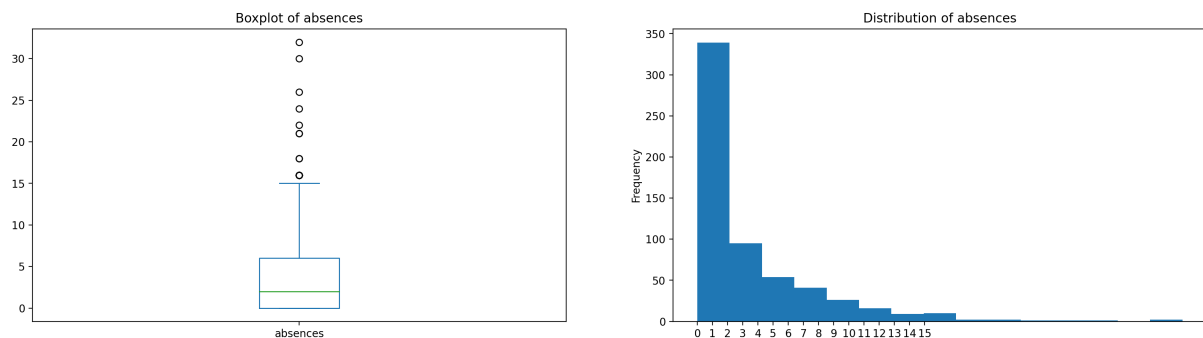


Abbildung 2.3: Verteilung der Abwesenheit in einem Boxplot (links) und einem Balkendiagramm (rechts)

## 2.3 Erste Korrelationsmatrix

Nachdem die Verteilungen verschiedener Spalten betrachtet wurden, kann die Korrelationsmatrix für den vorliegenden Datensatz bestimmt werden. Aus ihr können folgende Schlussfolgerungen gezogen werden:

1. Es gibt nur wenige starke Korrelationen mit den Schulnoten. Die größten haben die Noten untereinander, der Bildungsgrad der Eltern, die Durchfallquote, das Streben nach einem höheren Abschluss, die Schule, der Grund für die Schule und die Lernzeit.
2. Zwischen dem Beruf der Mutter und dem Vater, sowie ihrem Bildungsgrad besteht eine sehr hohe Korrelation, sodass der Beruf bzw. Bildungsgrad beider Elternteile zusammengefasst werden kann.
3. Zwischen dem täglichen und wöchentlichen Alkoholkonsum besteht ebenfalls eine hohe Korrelation, sodass sie sich ebenfalls zusammenfassen lassen.
4. Zwischen der Freizeit und dem Ausgehen mit Freunden besteht ebenfalls eine höhere Korrelation, die sich ebenfalls zusammenfassen lässt.



5. Die Spalten für die Schul- bzw. Elternunterstützung und der bezahlten Nachhilfe sind über binäre Skalen dargestellt, könnten aber zusammengefasst werden, weil sie ähnliche Sachverhalte beschreiben.

Auf Basis dieser Korrelationsmatrix wurde das erste Netz gebaut, das eine sehr lange Rechenzeit benötigte (für 120 Datenpunkte ca. 3 Minuten bei zwei fehlenden Spalten). Dadurch, dass die Berufe bzw. der Bildungsgrad doppelt auftreten und mit vielen anderen Knoten verbunden waren, mussten zu viele unterschiedliche Wahrscheinlichkeiten für das Bayes Netz berechnet werden. Um die Performance des Netzes zu verbessern, wurden die Spalten wie erwähnt zusammengefasst und es entstand die zweite Korrelationsmatrix.

## 2.4 Zweite Korrelationsmatrix

Für die zweite Korrelationsmatrix werden die Berufe der Eltern, ihr Bildungsgrad, der Alkoholkonsum, die Unterstützung und die freizeitlichen Aktivitäten jeweils in neue Spalten über verschiedene Methoden zusammengefasst. Die Berufe bekommen jeweils eine einzelne Spalte mit dem Wert 1 oder 0. Arbeitet mindestens einer der Elternteile in dem jeweiligen Beruf, dann wird der Wert 1 zugewiesen, wenn nicht, der Wert 0. Beim Alkoholverbrauch, den freizeitlichen Aktivitäten und dem Bildungsgrad der Eltern wird jeweils ein Median über die beiden Werte gebildet. Zuletzt wird die Unterstützung als Summe der drei Spalten *schoolsup*, *famsup* und *paid* gebildet.

Die daraus resultierende Korrelationsmatrix enthält ähnliche Korrelationen, wie die erste Matrix, enthält aber weniger Spalten. Dadurch hat das Bayes Netz weniger Knoten und benötigt weniger Rechenleistung bzw. Speicher.

## 3 Entwicklung und Evaluation

Die Entwicklung der Bayes Netze wurde im Notebook `src/model.ipynb` vorgenommen und dokumentiert. Weil im Verlauf der Arbeit verschiedene Netze und Ansätze entwickelt wurden, werden die einzelnen Fortschritte in diesem Kapitel genauer erläutert. Dabei wird zuerst der allgemeine Ablauf beschrieben und danach das große, sowie das getunte kleine Netz vorgestellt. Zum Schluss werden weitere Verbesserungsansätze erläutert und anhand der Genauigkeit und Fehlerraten evaluiert.

### 3.1 Allgemeiner Ablauf

Die Bayes Netze werden mit der Python-Bibliothek *pgmpy* erstellt. Dabei besteht die Möglichkeit Knoten und Kanten über das sogenannte *structure learning* automatisch zu lernen oder selbst die Kanten zu definieren. Danach lassen sich die Wahrscheinlichkeitstabellen über unterschiedliche Algorithmen aus den Daten lernen, sodass die Werte nicht von Hand berechnet werden müssen. Abschließend können auf Testdaten Vorhersagen getroffen werden und in einer Heatmap ausgewertet werden.

Bevor der ganze Prozess starten kann, müssen die Daten zuerst mit den aus der Datenanalyse gewonnenen Erkenntnisse vorverarbeitet werden. Hierfür wurde die Klasse *Preprocessing* geschrieben, die die Daten auf drei unterschiedliche Arten vorverarbeitet: normal, gruppiert (*tuned*) und binär. Die normale Vorverarbeitung ist die Grundlage und wendet die in der Datenanalyse vorgeschlagenen Begrenzungen für die Noten ( $<7$  und  $>17$ ), das Alter und die Abwesenheitstage an. Dadurch bildet es die erste Korrelationsmatrix und folglich das langsamere, größere Netz ab. Die getunte Vorverarbeitung wendet dann die vorgestellten Gruppierungen an und erzeugt somit weniger Knoten. Als letztes basiert die binäre Vorverarbeitung auf den getunten Daten und transformiert die Note G3 auf *bestanden* oder *nicht bestanden*, um später eine binäre Klassifikation abzubilden.

Nachdem die Netze für die jeweils vorverarbeiteten Daten erstellt und trainiert wurden, können sie evaluiert werden. Hierfür wurde eine Evaluationsklasse geschrieben, die die

Genauigkeit, den durchschnittlichen Fehler, den durchschnittlichen Fehler bei falscher Klassifikation, Präzision, Recall und F1-Wert berechnet. Zusätzlich werden die True bzw. False Positives und True bzw. False Negatives in einem Diagramm und die Konfusionsmatrix in einer Heatmap dargestellt. Dadurch können verschiedene Ansätze miteinander verglichen werden.

## 3.2 Großes Netz

Das große bzw. normale Netz basiert auf den normal vorverarbeiteten Daten, d. h., dass die Spalten äquivalent zum Ausgangsdatensatz sind und nur die binären Eigenschaften in 0 und 1 transformiert wurden. Auf Basis der ersten Korrelationsmatrix lässt sich dann, das in Abbildung 3.1 dargestellte Bayes Netz ableiten. Wichtig ist dabei, dass die Noten G1, G2 und G3 von den folgenden Knoten abhängen:

- Failures
- Studytime
- Higher
- Reason
- School
- Medu
- Fedu

Zusätzlich zeigen G1 und G2 auf G3, weil G3 die Zielausgabe ist. Durch die weiteren Kanten im Netz ist es möglich, dass z. B. der Knoten *studytime* fehlt, weil die Wahrscheinlichkeit aus *famsup* und *schoolsup* berechnet wird. Dennoch können einzelne Kanten anders interpretiert werden und das Netz ist ein möglicher Vorschlag von vielen probierten Kombinationen und unterscheidet sich folglich vom ersten implementierten Netz, auf das im Rahmen dieser Dokumentation nicht weiter eingegangen wird.

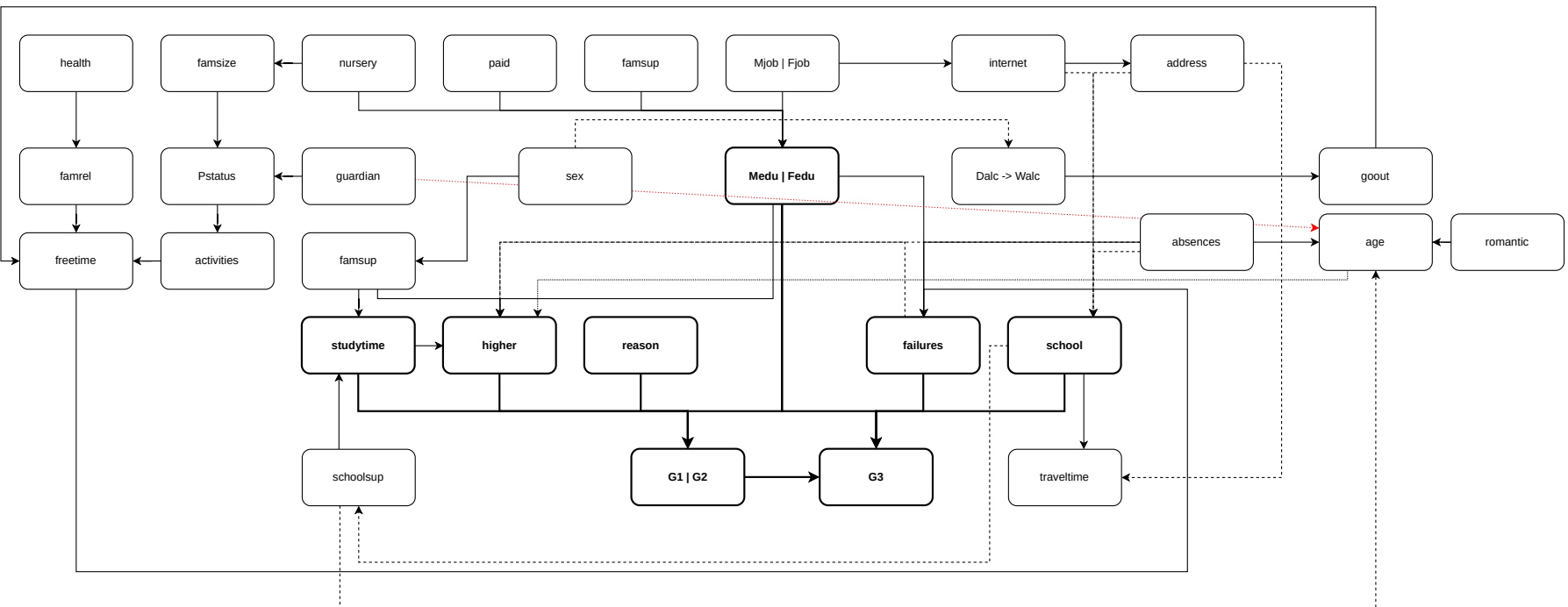


Abbildung 3.1: Großes Bayes Netz mit allen Knoten

### 3.3 Getuntetes Netz

Das getunte Netz verwendet die gruppierten Daten, sodass die Anzahl der Knoten kleiner ist als beim großen Netz. Hinzu kommt, dass die Anzahl der möglichen Kombinationen kleiner ist, weil kategorische Werte zusammengefasst wurden. Die Idee dahinter ist, dass weniger Knoten und Kombinationen gegen Overfitting helfen und das Netz leichter bzw. schneller wird. Nach dem Testen verschiedener Kanten ist das in Abbildung 3.2 dargestellte Netz entstanden. Die Knoten, die direkt auf die Note G3 zeigen, sind:

- Failures
- Studytime
- Reason
- School
- Higher
- G1
- G2

Auf G1 und G2 zeigen hingegen:

- Failures
- Studytime
- Reason
- School
- Higher
- Pedu

Dadurch, dass G1 und G2 zusätzlich die Kante mit *Pedu* haben, nutzt das Netz bereits auf dieser Ebene die gruppierten Daten und ist schneller als das große Netz. Weil aber die meisten anderen gruppierten Knoten an anderen Positionen im Netz sind, fällt die schnellere Rechenzeit erst auf, wenn mehrere Informationen fehlen.

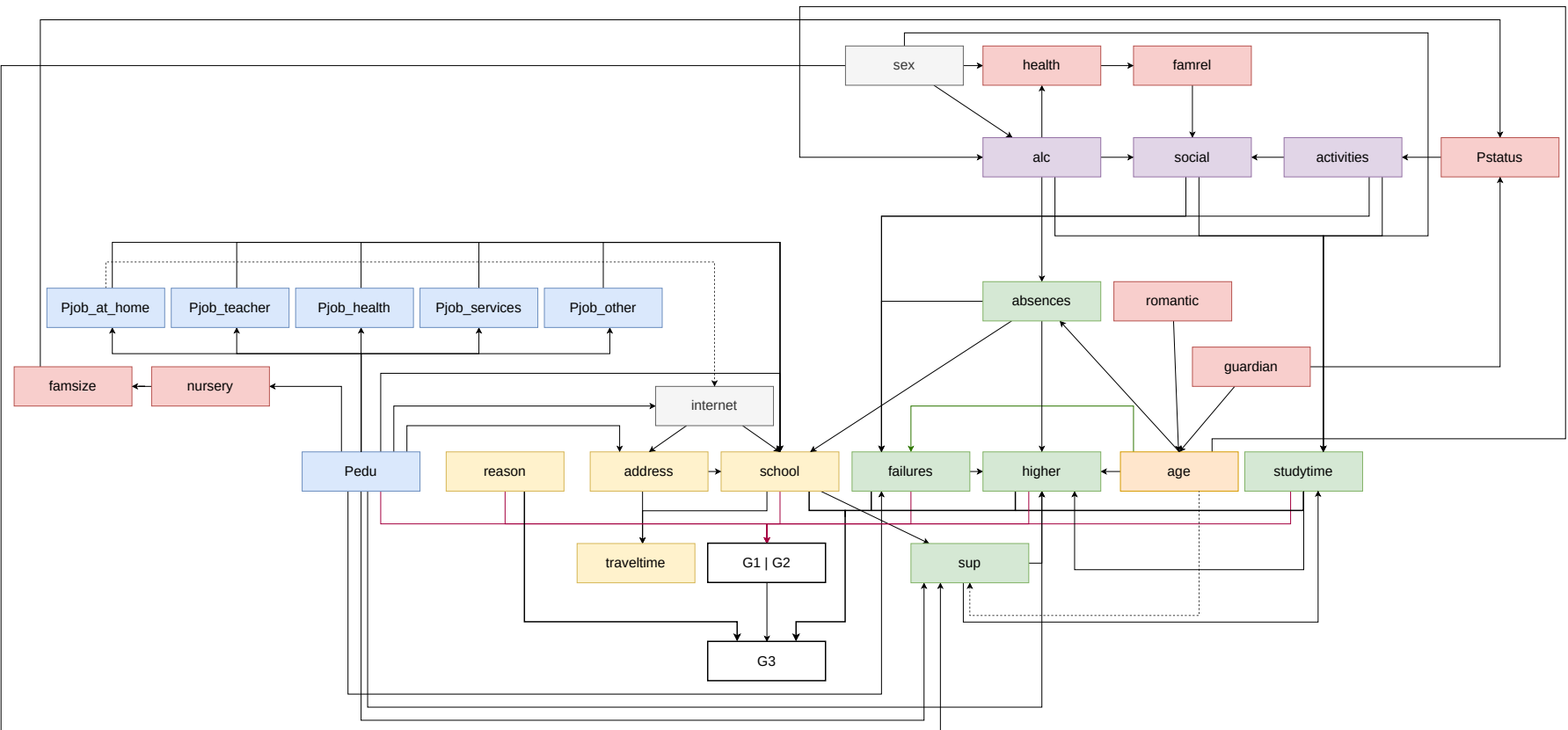


Abbildung 3.2: Getunttes Bayes Netz mit allen Knoten

## 3.4 Verbesserungsansätze

Im Rahmen dieses Projekts wurden weitere Ansätze entwickelt, um die Genauigkeit des Bayes Netzes zu verbessern. Der erste Ansatz ist das Entfernen von den in der Datenanalyse gefundenen Ausreißern aus den Daten. Dadurch soll das Netz näher an der echten Verteilung trainiert werden und bessere Ergebnisse erzielen.

Zusätzlich zu Ausreißern konzentriert sich der Datensatz stark auf die Noten zwischen 10 und 14, sodass viele Datenpunkte als diese Noten klassifiziert werden. In Kombination mit fehlenden oder unterrepräsentierten Noten kann das dazu führen, dass das Netz an den Trainingsdaten overfittet. Daraus folgt der zweite Verbesserungsansatz: Das Hinzufügen von Rauschen über eine Normalverteilung. Das Rauschen soll zu den Noten G1, G2 und G3 hinzugefügt werden, um das Problem zu verallgemeinern.

Zuletzt existiert im Ausgangsdatsatz eine undefinierte Spalte mit Werten zwischen 0 und 1, die als Gewichte verwendet werden können. Falls diese Spalte Gewichte darstellt, könnte so eine realistischere Verteilung gelernt werden und die Qualität der Ergebnisse besser werden.

## 3.5 Evaluation der Ansätze

Die vorgestellten Ansätze und Netze werden im letzten Schritt evaluiert und über ihre Konfusionsmatrix und Accuracy bzw. Fehlerraten verglichen. Tabelle 3.1 zeigt die Ergebnisse der verschiedenen Ansätze und Netze. Dabei ist das getunte Netz ohne Ausreißer insgesamt am besten, mit einer Genauigkeit von 22.2% bei fehlendem G1 und G2. Am schlechtesten schneidet das normale Netz mit einer Genauigkeit von 12% bei fehlendem G1 und G2 ab. Aus den Ergebnissen folgt, dass das zusätzliche Rauschen einen negativen und die Gewichte kaum einen Einfluss auf das Ergebnis haben.

Abgesehen davon, enthält Tabelle 3.1 eine grau markierte Zeile, bei der kein Train-Test-Split verwendet wurde, um die Performance auf den Trainingsdaten zu messen. Mit einer Genauigkeit von 43.6%, mit fehlenden G1 und G2 wäre dieses Ergebnis theoretisch das Beste. Weil es aber auf bereits bekannten Datenpunkte erzielt wurde, lässt es sich nicht mit den anderen Ergebnissen gleichsetzen und deutet darauf hin, dass das Netz leicht overfittet.

Setup	G1 fehlt	G2 fehlt	G1 und G2 fehlen
<b>Getunttes Netz</b> <b>Keine Gewichte</b> <b>Kein zusätzliches Rauschen</b> <b>Ohne Ausreißer</b> <b>80% Trainingsdaten</b> <b>20% Testdaten</b>	Accuracy: 0.368 Mean false error: 2.96 Mean absolute error: 1.87	Accuracy: 0.239 Mean false error: 3.22 Mean absolute error: 2.45	Accuracy: 0.222 Mean false error: 2.62 Mean absolute error: 2.03
Getunttes Netz Keine Gewichte Kein zusätzliches Rauschen Ohne Ausreißer <b>Trainingsdaten gleich Testdaten</b>	Accuracy: 0.774 Mean false error: 1.33 Mean absolute error: 0.3	Accuracy: 0.717 Mean false error: 1.72 Mean absolute error: 0.49	Accuracy: 0.436 Mean false error: 2.47 Mean absolute error: 1.39
Getunttes Netz Keine Gewichte Kein zusätzliches Rauschen <b>Mit Ausreißer</b> 80% Trainingsdaten 20% Testdaten	Accuracy: 0.383 Mean false error: 3.18 Mean absolute error: 1.96	Accuracy: 0.217 Mean false error: 3.09 Mean absolute error: 2.42	Accuracy: 0.15 Mean false error: 2.89 Mean absolute error: 2.46
<b>Normales Netz</b> Keine Gewichte Kein zusätzliches Rauschen Ohne Ausreißer 80% Trainingsdaten 20% Testdaten	Accuracy: 0.103 Mean false error: 5.51 Mean absolute error: 4.95	Accuracy: 0.051 Mean false error: 5.47 Mean absolute error: 5.19	Accuracy: 0.12 Mean false error: 3.75 Mean absolute error: 3.3
Getunttes Netz Keine Gewichte <b>Zusätzliches Rauschen</b> Ohne Ausreißer 80% Trainingsdaten 20% Testdaten	Accuracy: 0.188 Mean false error: 2.92 Mean absolute error: 2.37	Accuracy: 0.154 Mean false error: 3.15 Mean absolute error: 2.67	Accuracy: 0.162 Mean false error: 2.49 Mean absolute error: 2.09
Getunttes Netz <b>Zusätzliche Gewichte</b> Kein zusätzliches Rauschen Ohne Ausreißer 80% Trainingsdaten 20% Testdaten	Accuracy: 0.333 Mean false error: 2.86 Mean absolute error: 1.91	Accuracy: 0.222 Mean false error: 3.13 Mean absolute error: 2.44	Accuracy: 0.179 Mean false error: 2.52 Mean absolute error: 2.07

Tabelle 3.1: Ergebnisse der verschiedenen Ansätze

Zusätzlich zur Genauigkeit des Netzes, lassen sich die Konfusionsmatrix vergleichen, um Fehlklassifikationen zu erkennen. In den Abbildungen 2.3 bis 2.5 werden die Konfusionsma-



trizen des getunten Netzes mit (Abbildung 2.3) bzw. ohne (Abbildung 2.4) Train-Test-Split und des großen Netzes (Abbildung 2.5) dargestellt.

In Abbildung 2.3 fällt auf, dass die hauptsächliche Fehlerursache die Note  $<7$  ist. Das könnte daran liegen, dass pgmpy bei unbekannten Kombinationen, die nicht von den Trainingsdaten abgebildet werden, von einer Gleichverteilung ausgeht. Dadurch, dass  $<7$  der erste Wert in dieser Verteilung ist, wird bei der Klassifizierung auf den ersten Wert zurückgegriffen. Betrachtet man 3.4a, dann wird diese Theorie bestärkt, weil hier alle Kombinationen abgebildet sind.

Abgesehen davon sieht man in 3.3c, dass viele Werte auf 11 und 13 klassifiziert werden, was vermutlich an der grundlegenden Verteilung liegt. Denn aus der Datenanalyse folgt, dass die Note 11 am meisten vertreten ist.

Wird als letztes Abbildung 2.5 betrachtet, dann fällt auf, dass hier ebenfalls die Klassifikation auf  $<7$  das Problem ist. Doch im Unterschied zum getunten Netz werden hier die Noten aus mehr verschiedenen Zuständen berechnet und daher treten mehr unbekannte Kombinationen auf.

Abschließen folgt aus der Evaluation, dass das getunte Netz einen akzeptablen Klassifikator für den grundlegenden Datensatz liefert. Dennoch ist der Klassifikator nicht perfekt, weil er 14 verschiedene Zustände klassifizieren soll, wobei sich nicht alle Kombinationen abdecken lassen. Dazu kommen, die geringen Korrelationen im Datensatz, die die Bestimmung der Netzstruktur erschweren. Vergleicht man den Klassifikator mit anderen Netzstrukturen, dann fällt aber auf, dass die Klassifikationsergebnisse, den Umständen entsprechend gut sind. Daraus folgt, dass er für die entwickelte Anwendung verwendet werden kann, um Schulnoten von portugiesischen Schülern zu klassifizieren.

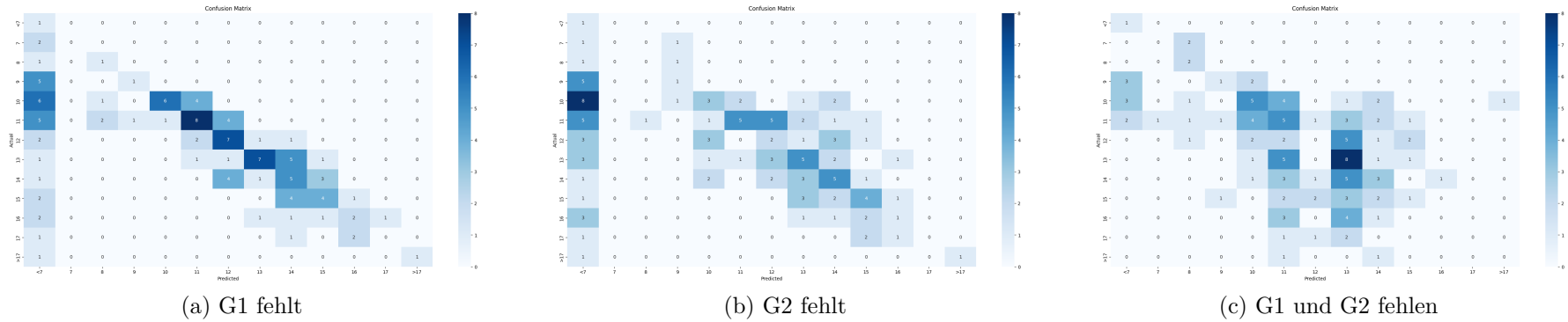


Abbildung 3.3: Konfusionsmatrizen des besten Ansatzes

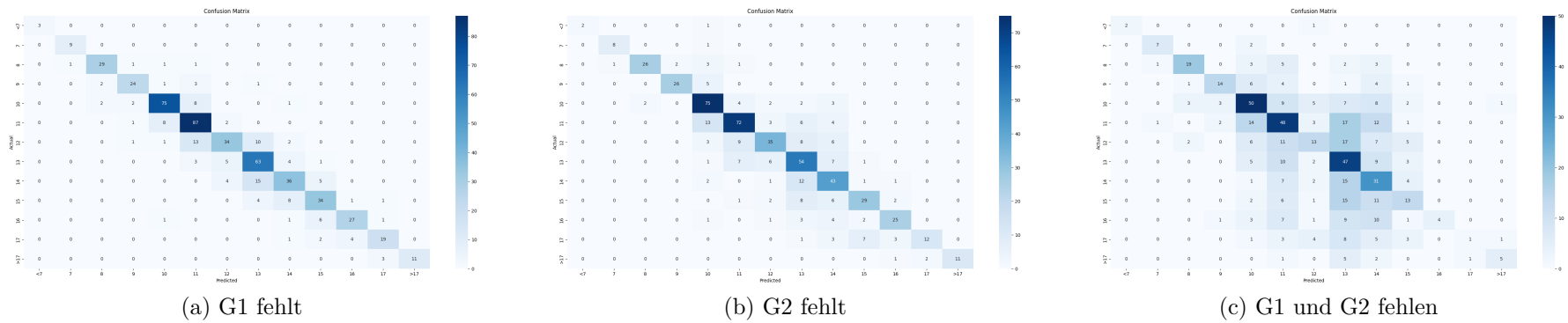


Abbildung 3.4: Konfusionsmatrizen des getunten Netzes ohne Train-Test-Split

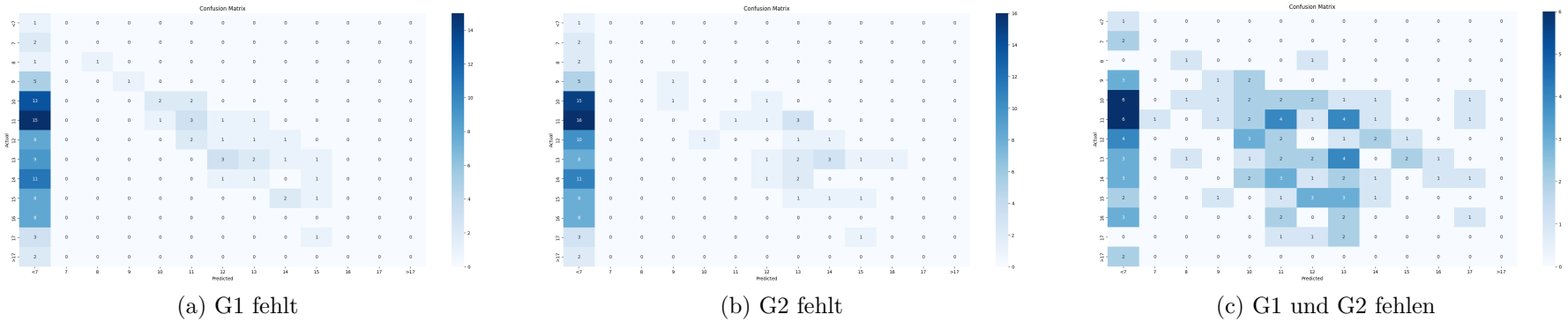
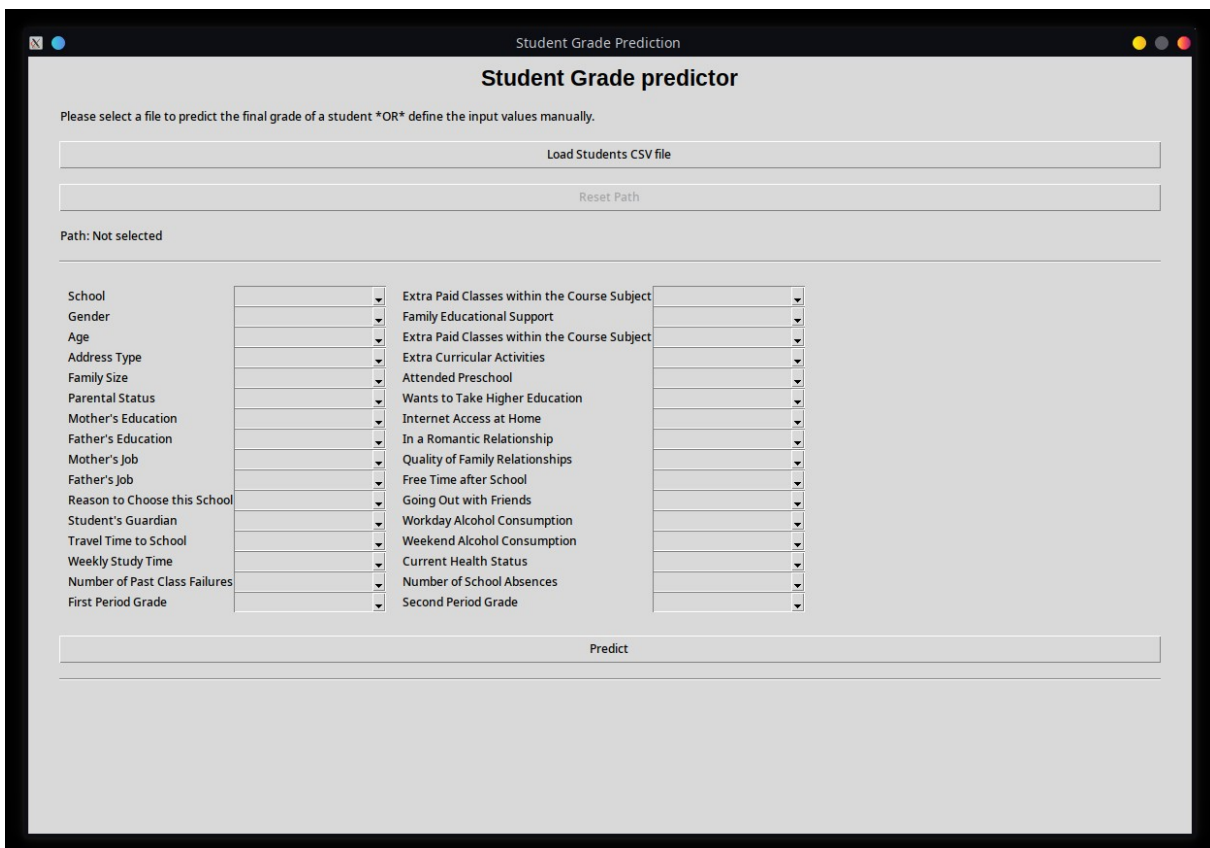


Abbildung 3.5: Konfusionsmatrizen des großen Netzes

## 4 Anwendung

Teil des Projekts ist die Entwicklung einer Anwendung, um die Noten von portugiesischen Schülern mithilfe von Bayes Netzen zu klassifizieren. Hierfür wurde in `src/app.ipynb` eine *Tkinter*-Anwendung implementiert, die entweder eine CSV Datei importieren und klassifizieren kann oder die Daten aus einem Formular liest und klassifiziert.

Abbildung 4.1 zeigt das Anwendungsfenster, mit dem Import im oberen Bereich und dem Formular in der Mitte der Anwendung. Wird eine Datei importiert, dann werden die Noten für alle Einträge vorhergesagt und in eine neue Datei exportiert. Werden die Werte hingegen von Hand eingetragen, dann wird die vorhergesagte Note unter dem *predict* Knopf angezeigt.



Student Grade Prediction

### Student Grade predictor

Please select a file to predict the final grade of a student \*OR\* define the input values manually.

Load Students CSV file

Reset Path

Path: Not selected

School		Extra Paid Classes within the Course Subject	
Gender		Family Educational Support	
Age		Extra Paid Classes within the Course Subject	
Address Type		Extra Curricular Activities	
Family Size		Attended Preschool	
Parental Status		Wants to Take Higher Education	
Mother's Education		Internet Access at Home	
Father's Education		In a Romantic Relationship	
Mother's Job		Quality of Family Relationships	
Father's Job		Free Time after School	
Reason to Choose this School		Going Out with Friends	
Student's Guardian		Workday Alcohol Consumption	
Travel Time to School		Weekend Alcohol Consumption	
Weekly Study Time		Current Health Status	
Number of Past Class Failures		Number of School Absences	
First Period Grade		Second Period Grade	

Predict

Abbildung 4.1: Tkinter Anwendung

Im Hintergrund verwendet die App das in Kapitel 3 vorgestellte getunte Netzwerk, ohne Ausreißer in den Trainingsdaten. Für jeden abgefragten Datensatz werden die Werte durch das Netz bestimmt, sodass sich fehlende Werte bestimmen lassen und eine Note für den Schüler vorhergesagt werden kann.

## 5 Zusammenfassung und Ausblick

In diesem Projekt wurde ein akzeptables Netz für die Klassifikation von portugiesischen Schulnoten entwickelt, dass bei fehlenden G1 und G2 eine Genauigkeit von ca. 22.2% aufweist. Das Netz ist das Ergebnis einer ausführlichen Datenanalyse (Kapitel 2), aus der eine umfangreiche Datenvorverarbeitung folgt. Zusätzlich hat das Netz zu viele Knoten mit zu geringen Korrelationen, damit das Lernen einer Struktur funktioniert. Deswegen wurden im Rahmen des Projekts verschiedene von Hand implementiert Netze ausprobiert, wobei das vorgestellte Netz die beste Genauigkeit und geringste Fehlerrate hat.

Zusätzlich wurden weitere Optimierungen vorgestellt, um die Genauigkeit zu steigern. Dabei wird in der Evaluation (Kapitel 3) klar, dass zusätzliches Rauschen oder Kantengewichte keinen positiven Einfluss auf das Ergebnis haben.

Abschließend wird das getunte Netz für die Python Anwendung zur Klassifikation von portugiesischen Schulnoten verwendet. Damit die Ergebnisse in Zukunft eine höhere Genauigkeit aufweisen, könnten mehr Daten erhoben werden. Das können entweder mehr Schüler oder mehr Eigenschaften sein, um die Klassifikation besser durchzuführen. Alternativ zu Bayes Netzen können auch andere maschinelle Lernverfahren ausprobiert werden und ihre Ergebnisse lassen sich dann mit den hier vorgestellten Bayes Netzen vergleichen.

# Anhang

## Literatur

- [1] P. Cortez und A. Silva, „Using Data Mining to Predict Secondary School Student Performance,“ *A. Brito and J. Teixeira (Eds.) Proceedings of 5th FUTURE BUSINESS TECHNOLOGY Conference*, S. 5–12, Apr. 2008, ISSN: ISBN 978-9077381-39-7.
- [2] D. Reichardt, *Programmmentwurf Künstliche Intelligenz - Bayes Netze*, de-DE, März 2023.