INSTITUT FÜR INFORMATIK
Datenbanken und Informationssysteme

Universitätsstr. 1 D–40225 Düsseldorf

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Vergleich von Machine Learning-Verfahren für das Author Profiling

Zeljko Bekcic

Bachelorarbeit

Beginn der Arbeit:	03. Juni 2018
Abgabe der Arbeit:	17. September 2018
Gutachter:	Prof. Dr. Stefan Conrad
	Prof. Dr. Gunnar Klau

Erklärung

Hiermit versichere ich, dass ich diese Bachelorarbeit selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Düsseldorf, den 17. September 2018

Zeljko Bekcic

Zusammenfassung

Author Profiling beschreibt die Klassifizierung von Texten bezüglich Merkmalen der Autoren. Hierfür wurde ein Datensatz welcher auf Twitter gesammelt wurde verwendet. Anschließend wurden die Algorithmen aus dem Machine Learning in klassische Machine-Learning Algorithmen und Deep-Learning Algorithmen getrennt. Diese wurden erstmals voneinander isoliert betrachtet und anschließend wurde der Deep-Learning Ansatz genutzt um zu beobachten, ob dieser die Dimensionalität des Inputs reduzieren kann. Bevor jedoch die Algorithmen miteinander verglichen wurden, wird der Datensatz untersucht und die mit dem Datensatz verbundenen Probleme angesprochen. Bei der Untersuchung des Datensatzes wird festgestellt, dass die Daten nicht balanciert sind. Daher müssen die Algorithmen anders ausgewertet werden. Anschließend wird beschrieben, wie der Datensatz verarbeitet wurde und in Form von Zahlen für die Algorithmen dargestellt wird. Hiernach wird erläutert, welche Algorithmen aus dem klassischen Machine-Learning und Deep-Learning miteinander verglichen werden. Des Weiteren wird dabei beschrieben, wie die Hyperparameter für diese Algorithmen bestimmt wurden. Zuletzt werden die Ergebnisse evaluiert. Es werden hierbei die klassischen Machine-Learning Algorithmen mit und ohne Featue Engineering bezüglich den Merkmalen der Autoren ausgewertet. Des weiteren werden die neuronalen Netze ebenfalls mit den Daten auf dem Datensatz evaluiert. Anschließend wird untersucht, wie die neuronalen Netze sich für die reduktion der Dimensionalität der Daten eignen. Hierfür werden die Daten anschließend dem klassischen Machine-Learning Algorithmus übergeben.

Inhaltsverzeichnis

1 Einleitung

In diesem Kapitel wird zunächst erklärt was Author Profiling ist und in welchen Anwendungsfällen es relevanz findet. Anschließend wird erläutert, was das Ziel in dieser Bachelorarbeit ist.

1.1 Motivation

Author Profiling ist ein Klassifizierungsproblem, in welchem der Input für Machine-Learning Algorithmen Texte sind und es nach Merkmalen von Autoren klassifiziert wird. Diese Texte können beispielsweise Nachrichten sein, die von sozialen Netzwerken wie Twitter oder Facebook stammen. Somit wird durch den Text klar, welches Geschlecht der Autor hat oder welcher Altersgruppe er angehört.

Diese Problemstellung hat unter anderem Relevanz im Marketing und in der Kriminalistik. Im Marketing kann Author Profiling genutzt werden, um eine Analyse der Kunden durchzuführen. Hierbei werden Bewertungen oder Kommentare aus dem Internet nach gewünschten Merkmalen klassifiziert. Damit können dann gewisse Zielgruppen besser angesprochen werden. Durch die Arbeit von Lin (2010) wird außerdem deutlich, dass Menschen mit einer extrovertierten Persönlichkeit loyaler gegenüber Marken sind als introvertierte Menschen. In der Kriminalistik kann unter den Tatverdächtigen der Autor eines Textes bestimmt werden. Diese beiden Anwendungsbeispiele stammen aus „Overview of the Author Profiling Task at PAN 2013“ von Pardo et al. (2013). Passend zu dem letzten Beispiel hat die Natural Language Processing (NLP) Forschungsgruppe von Cambridge einen wissenschaftlichen Artikel veröffentlicht in dem mittels Author Profiling Belästigungen wie Rassismus oder Sexismus in Nachrichten festgestellt wurde (Mishra et al., 2018).

Es folgt aus dem No Free Lunch Theorem von Wolpert und Macready (1995, S. 67–82), dass Machine-Learning Verfahren sich unterschiedlich auf Problemstellungen verhalten. Daher ist es naheliegend, dass bestimmte Verfahren besser als andere abschneiden. Zusätzlich gibt es noch verschiedene Möglichkeiten den Text als Zahlen zu repräsentieren, hierbei können beispielsweise können Embeddings gewählt werden um den Text als Vektoren darzustellen. Unter anderem kann zu Word Embeddings gegriffen werden, diese Embeddings liefern neben der Vektordarstellung des Wortes auch die Kontextinformation. Diese Darstellungen der Texte werden dann dem Machine-Learning Algorithmus übergeben werden. Die Wahl einer geeigneten Methode den Text zu vektorisieren kann eine Rolle spielen, denn eine schlechte Repräsentation des Textes in Zahlen die Qualität des Algorithmus negativ beeinflussen.

1.2 Zielsetzung der Bachelorarbeit

Das Ziel dieser Bachelorarbeit ist es zu untersuchen, wie gut verschiedene Machine-Learning Verfahren auf dem Auhor Profiling Problem abschneiden. Hierbei wird zwischen klassischen Machine-Learning Ansätzen und den Deep-Learning Ansätzen getrennt. Es werden bei den klassischen Machine-Learning Verfahren unter anderem Algo-

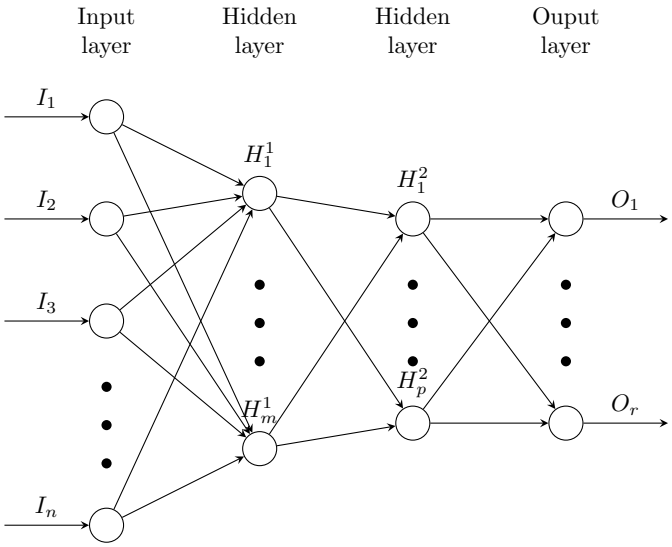

Abbildung 1: Ein *Fully-Connected Neural Network* mit vier Layern

rithmen wie Decision Trees (L. Breiman et al., 1984) oder Support Vector Machines (SVM) (Cortes und Vapnik, 1995, S. 273–297) betrachtet. Bei dem Deep-Learning Ansatz werden verschiedene Architekturen verglichen. Es werden von einfacheren Fully Connected Neural Networks bis hin zu komplexere Architekturen wie zum Beispiel Long Short-Term Memory Neural Networks (Hochreiter und Schmidhuber, 1997) betrachtet. Im Anschluss werden diese beiden Ansätze miteinander kombiniert. Es wird ein neuronales Neutz genommen und geschaut, ob es in den Layern eine niedrigdimensionale Darstellung der Daten bieten kann welche eine höhere Informationsgüte besitzt, die sich der Algorithmus aus dem klassischen Machine-Learning zunutze machen kann. Das wird mit den Abbildungen 1 und 2 veranschaulicht. In 2 wird das veranschaulicht. Hier werden die ersten zwei Layer des neuronalen Netzes aus Abbildung 1 verwendet um die Dimensionalität des Inputs zu verkleinern. Anschließend wird diese Darstellung dem klassischen Machine-Learning Algorithmus übergeben.

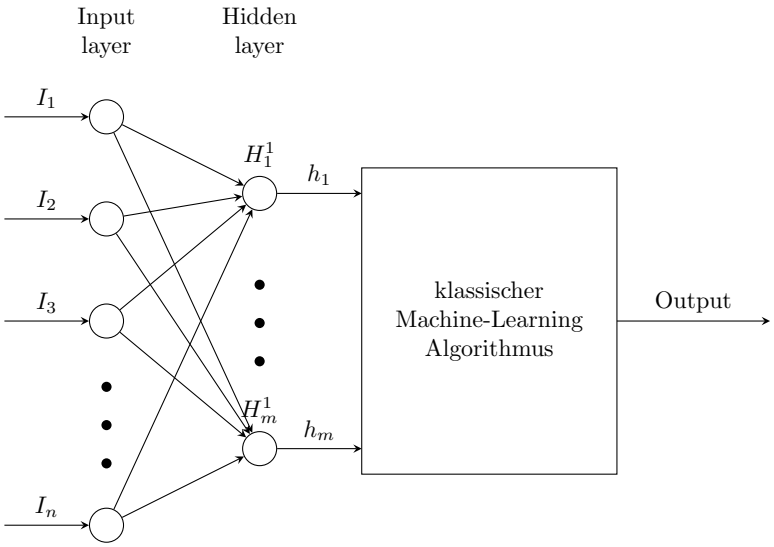

Abbildung 2: Ein klassischer Machine-Learning Algorithmus macht sich das Ergebniss zwischen den Layern für die Klassifizierung zunutze.

2 Datensatz

Dieses Kapitel wird auf den Datensatz genauer eingehen. Vorerst wird beschrieben wie der Datensatz gesammelt wurde und welche Problematik mit der damit Verbundenen Plattform entsteht. Darauf folgend wird der Datensatz visualisiert und letztendlich die Verarbeitung der Daten thematisiert um den Datensatz für die Machine-Learning Alogrithmen zu verwenden.

2.1 Der TwiSty Datensatz

In dieser Bachelorarbeit wird der TwiSty Korpus (Verhoeven et al., 2016) verwendet, ein Datensatz welcher für Author Profiling erstellt wurde. Dieser Korpus besteht aus Nachrichten von Twitter, welche Tweets genannt werden, in welchem jede Nachricht mit dem Geschlecht und dem Myers-Briggs Type Indicator (MBTI) des Autors gelabelt wurde (Verhoeven et al., 2016). Der Myers-Briggs Type Indicator ist ein Ansatz den Menschen in eine von insgesamt 16 Kategorien einzuordnen, in Abhängigkeit davon, wie wir mit unserem Umfeld interagieren (vgl. Foundation (2018)). Dieser Wert setzt sich aus insgesamt 4 Zeichen zusammen, die Bedeutung der Zeichen werden in Tabelle 1 dargestellt. Beispielsweise ist ISTP ein gültiger MBTI.

Des Weiteren bietet der Korpus verschiedene Sprachen. Diese sind Niederländisch, Deutsch, Französisch, Italienisch, Portugiesisch und Spanisch. Hier wird der deutschsprachige Teil des Datensatzes für diese Problemstellung verwendet.

	Mögliche Werte	Bedeutung
1. Zeichen	I, E	Introvertiert bzw. Extrovertiert
2. Zeichen	S, N	Sensorisch bzw. Intuitiv
3. Zeichen	T, F	Denkend bzw. Fühlend
4. Zeichen	P, J	Wahrnehmend bzw. Urteilend

Tabelle 1: Ein Überblick über die Werte aus denen sich der MBTI zusammensetzt.

2.2 Datamining des Datensatzes

Twitter ist eine Plattform welche viele Daten, aufgrund ihrer hohen Nutzeranzahl und Aktivität dieser Nutzer, aggregiert. Daher bietet es sich sehr an, Datensätze mithilfe von Twitter zu erstellen, denn alle Tweets sind hier öffentlich und können somit einfacher gesammelt werden. In Verhoeven et al. (2016) wird zuerst nach Nutzer gesucht, welche Auskunft über ihren MBTI Wert geben und anschließend wurde von diesen Nutzern das Geschlecht bestimmt. Darauf folgend erhielten Verhoeven et al. (2016) von diesen Nutzern die Tweets gesammelt und anschließend nach einer Sprache klassifiziert.

Das Problem welches Twitter mit sich bring ist, dass wenn Tweets gesammelt werden und in Form eines Datensatzes veröffentlicht werden sollen, dürfen nur die TwitterIDs zu Verfügung gestellt werden (Twitter, 2018). Das kann einerseits als Vorteil aufgefasst werden, denn es ist einfacher für den Nutzer Kontrolle über die Daten zu behalten. Sobald der Tweet auf Twitter gelöscht wird, ist dieser Tweet nun nicht mehr im Datensatz enthalten. Andererseits ist genau dieser Punkt problematisch. Es kann über die Zeit passieren, dass Tweets und/oder Nutzeraccounts gelöscht werden, somit ist dieser Datensatz nicht persistent und die Ergebnisse ggf. nicht mehr reproduzierbar und genau das ist das Problem beim TwiSty Datensatz. Ein weiterer Kritikpunkt ist die Bestimmung des MBTI. Hierbei wurde wie erwähnt nach Tweets gesucht, welche Effektiv etwas über den MBTI aussagen. Jedoch wurde nicht überprüft, ob dieser MBTI zu dem Nutzer passt. Somit kann es sich um einen vom Nutzer selbst bestimmten Wert handeln und dieser kann vom Tatsächlichen abweichen.

2.3 Ein schon gesammelter Datensatz

Für diese Bachelorarbeit wurde der TwiSty-Korpus mit Tweets zur Verfügung gestellt.

2.4 Visualisierung des Datensatzes

Im Folgenden wird der Datensatz visualisiert und betrachtet, ob besondere Vorkommnisse vorzufinden sind, wie zum Beispiel ein Zusammenhang zwischen dem Geschlecht bzw. dem MBTI und bestimmten Features. Es wird zuerst das Geschlecht und dann der MBTI Wert der Nutzer bezüglich der Anzahl der Tweets betrachtet. Anschließend werden die Geschlechter und die MBTI Werte zusammen bezüglich der Anzahl der Tweets visualisiert. In Abschnitt 2.7 wird dann detailierter auf den Inhalt der Tweets eingegangen.

Insgesamt beinhaltet der Datensatz 194 Autoren, mit 208159 Tweets. Hiervon sind 94 weiblich und 98 männlich. Die Daten weisen auf, dass die beiden Geschlechter eine ungefähr gleiche Anzahl an Tweets verfasst haben, jedoch sind die männlichen Teilnehmer etwas aktiver auf Twitter und haben daher etwas mehr Tweets veröffentlicht. Konkret sind von den weiblichen Teilnehmern 96199 Tweets verfasst worden und 111960 von den männlichen Teilnehmern. Das ergibt eine Differenz von 15761 Tweets, welche noch vernachlässigbar ist (siehe Abbildung 3).

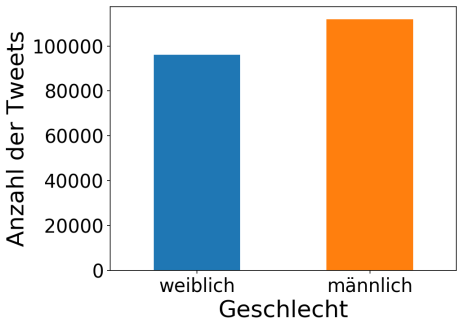

Abbildung 3: Anzahl der Tweets von jedem Geschlecht.

Hiernach wird die Verteilung der Nutzer auf die MBTIs untersucht. Es wird durch die Abbildung 4 deutlich, dass Teilnehmer mit dem MBTI 'INFP', 'INFJ' und 'INTP' am meisten vertreten sind, vorallem betrifft das 'INFP'. Auffällig ist hierbei, dass die drei am meisten vertreteten Gruppen hier alle einen MBTI haben welcher mit 'IN' beginnt. Des Weiteren fällt auf, dass die drei am wenigsten vertreteten Gruppen 'ESTJ', 'ESTP' und 'ESFP' sind und deren MBTI mit 'ES' beginnt. Wenn der MBTI beispielsweise auf das erste Zeichen eingeschränkt wird, welches angibt ob eine Person introvertiert bzw. extrovertiert ist wird deutlich, dass auf Twitter mehr introvertierte als extrovertierte Teilnehmer vorzufinden sind (siehe Abbildung 5). Ähnliche Verteilungen sind auch bei den anderen Werten des MBTI zu beobachten.

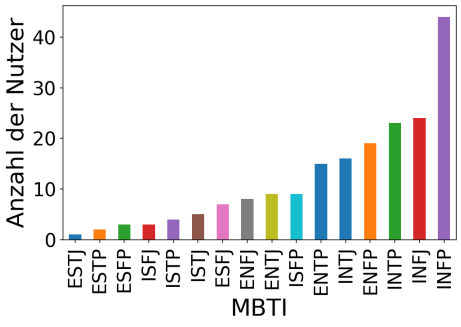

Abbildung 4: Anzahl der Nutzer von jedem MBTI.

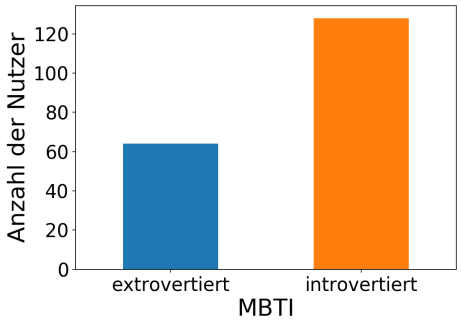

Abbildung 5: Anzahl der Nutzer von den introvertierten/extrovertierten Nutzern.

Nun wird der Blick auf die Anzahl der Tweets von den einzelnen MBTI Kategorien gelenkt. Die Anzahl der Nutzer in den einzelnen Kategorien gibt noch keine Aussage über

die Anzahl der Tweets. Es kann sein, dass ein Nutzer in der MBTI Kategorie welche am schwächsten vertreten ist, immernoch mehr Tweets verfasst als Nutzer deren MBTI deutlich mehr Verteten ist. Abbildung 6 zeigt, dass eine schwächer vertretete MBTI Kategorie seltener mehr Tweets verfasst als eine stärker Vertretene MBTI Kategorie erstellt hat. In Abbildung 7 ist erneut nach introvertierten und extrovertierten Teilnehmern unterschieden worden. Hier wird nochmals deutlich, wie groß der Unterschied zwischen in der Anzahl der Tweets ist. Die introvertierten Teilnehmer haben insgesamt 135710 Tweets verfasst, wohingegen die extrovertierten Teilnehmer fast halbsoviel getweetet haben, nämlich 72449 mal. Dies führt dazu, dass die Qualität der Machine-Learning Algorithmen anders gemessen werden müssen.

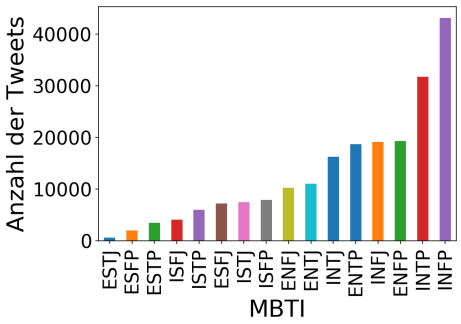

Abbildung 6: Anzahl der Tweets von jedem MBTI.

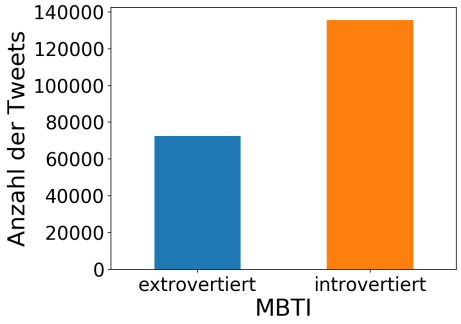

Abbildung 7: Anzahl der Tweets von den introvertierten/extrovertierten Nutzern.

Es ist in Abbildung 8 erkennbar, dass ein Geschlecht in den einzelnen MBTIs dominant präsent ist bezüglich der Anzahl der Nutzer. Selbiges in der Anzahl der Tweets in Abbildung 9 zu beobachten.

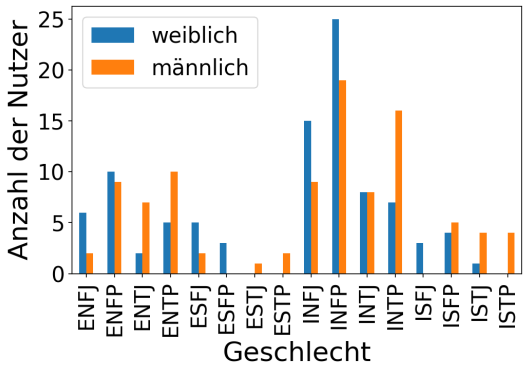

Abbildung 8: Anzahl der Nutzer von jedem MBTI aufgeteilt auf die Geschlechter.

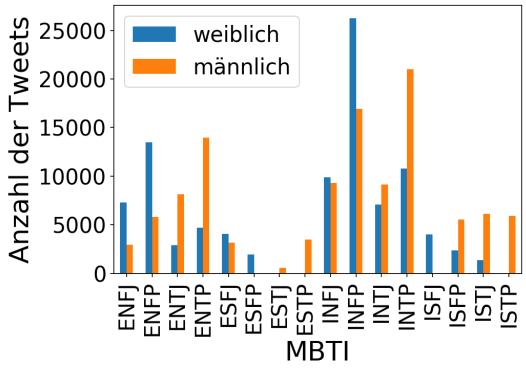

Abbildung 9: Anzahl der Tweets von jedem MBTI.

Jedoch ist anhand der beiden Abbildungen 10 und 11 zu entnehmen, dass die Anzahl der extrovertierten bzw. introvertierten der weiblichen und männlichen Teilnehmer ungefähr gleich ist.

Abschließend zu dem Überblick über den Daten kann erstmals folgendes Fazit gezogen

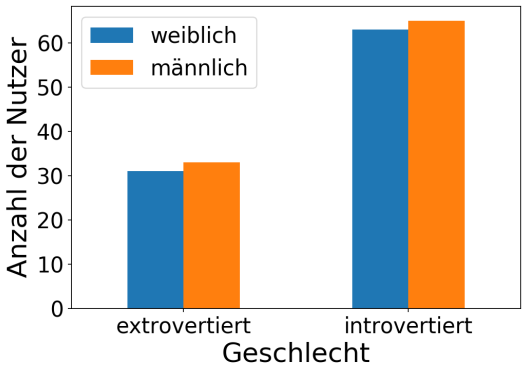

Abbildung 10: Anzahl der extrovertierten/introvertierten Nutzer aufgeteilt auf die Geschlechter.

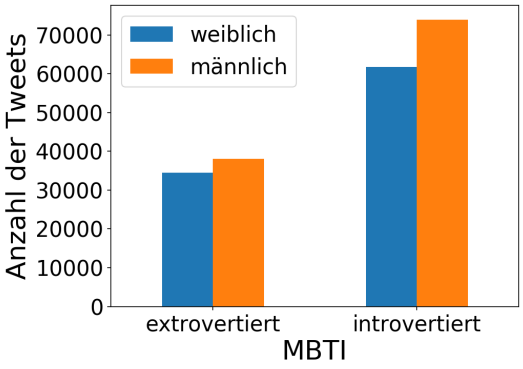

Abbildung 11: Anzahl der Tweets von jedem Gechlecht aufgeteilt nach den extrovertierten/introvertierten Nutzern.

werden: Die Geschlechter sind bezüglich den Daten relativ gut aufgeteilt, bis auf den geringen Teil mehr bei den männlichen Autoren sind die Datenteile gleichgroß. Hingegen sieht es bei dem MBTI anders aus. Hier sind die Daten nicht gleichmäßig verteilt. Es sind vorallem einige Klassen nicht gut genug vertreten, sodass Oversampling nicht sinnvoll wäre. Eine Lösung für dieses Problem ist es, sich auf ein Zeichen des MBTI zu konzentrieren, beispielsweise das erste. Dieses gibt an, ob der Autor extrovertiert oder introvertiert ist. Zwar gibt es von den introvertierten Autoren immernoch ca. doppelt soviele Tweets wie von den extrovertierten Autoren, jedoch sind hier erstmals genug Tweets von diesen Autoren vorhanden um danach zu klassifizieren.

2.5 Preprocessing der Daten

Die Tweets liegen in Form von JSON-Dateien vor, von diesen Dateien werden die Textnachrichten extrahiert um diese anschließend wie folgt zu preprocessen.

1. Text in Kleinbuchstaben überführen

   Auf sozialen Netzwerken wird häufig nicht auch die Groß-/Kleinschreibung geachtet, daher kann das im Allgemeinen vernachlässigt werden.

2. HTML-Entities der Textnachrichten ausflösen

   In den Tweets sind beispielsweise Umlaute als HTML-Entities repräsentiert worden. Damit in den folgenden Schritten keine Komplikationen entstehen wurden diese direkt am Anfang zu dem eigentlich zu repräsentierenden Symbol aufgelöst.

3. Mithilfe von spaCy die Nachrichten tokenizieren

   SpaCy [1] ist ein zuverlässiges Werkzeug mit einer umfangreichen Anzahl an Funktionen für Natural Language Processing (NLP). Beispielsweise kann es neben Tokenization auch die POS-Tags und Dependenzen von Wörtern bestimmen. Es bietet

[1] https://spacy.io/

des Weiteren viele verschiedene Sprachmodelle, darunter befindet sich auch ein deutsches Sprachmodell, was für den gewählten Datensatz passend ist.

4. Stoppwörter entfernen

   Stoppwörter sind häufig vorkommende Wörter, diese können ggf. keine besonderen Informationen über den Autor enthalten.

5. Interpunktionen, URLs, Sonderzeichen und Zahlen aus dem Text entfernen

   Diese Elemente werden im Text keine Informationen über das linguistische Profil der Nutzer liefern.

6. Zu kurze Token entfernen

   Tokens welche eine Länge 1 oder weniger haben werden sind zu kurz und wurden ebenfalls aus dem Datensatz herausgenommen, denn es sind keine Wörter mit nur einem Buchstaben in der deutschen Sprache. Ebenfalls wenn spaCy etwas nicht als Interpunktion, URL, Sonderzeichen oder Zahl Kategorisiert hat kann es hiermit aus dem Datensatz genommen werden.

7. Leere Strings entfernen

   Zuletzt werden leere Strings aus dem Datensatz genommen, denn diese können nicht sinnvoll mit einer der in Abschnitt 2.6 genannten Methoden als Vektoren dargestellt werden.

2.6 Vektorisierung des Textes

Es wurde schon angemerkt, dass die Wahl des Vektorisierung des Textes eine Rolle spielen kann, wie gut der Machine-Learning Algorithmus auf den Daten klassifiziert. Nun werden die hier verwendeten Embeddings betrachtet, welche für den Vergleich verwendet werden.

- Bag Of Words

  Ist ein sehr einfache Methode einen Text zu Vektorisieren. Jedoch werden keine Information über den Kontex der Wörter behalten. Denn die Reihenfolge der Wörter bleibt in der Vektordarstellung nicht erhalten.

  Beispiel: Die folgenden Sätze werden folgendermaßen dargestellt werden.

  "Das ist ein Stift und ein Blatt Papier." $\rightarrow (1 \quad 1 \quad 2 \quad 1 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0)$

  "Das ist ein Hund und keine Katze." $\rightarrow (1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1)$

  Es ist direkt zu erkennen, dass aus dem Vektor die Bedeutung des zweiten Satzes nicht mehr eindeutig ist.

- word2vec (Mikolov et al., 2013)

  Word2vec ist ein Word-Embedding welches den Kontext der Wörter mitbeachtet um Ähnlichkeiten von Wörtern festzustellen. Dieser Ansatz bietet sich für Author Profiling an, denn nun liegen semantisch ähnliche Wörter näher zueinander im Vektorraum. Somit können nun bestimmte Wortgruppen erfasst werden.

- GloVe (Pennington et al., 2014)

  Word2vec und GloVe sind Embeddings welche durch neuronale Netze erstellt werden. Diese word2vec wird supervised trainiert. GloVe ist ebenfalls ein Embedding welches durch ein neuronales Netz erstellt wird. Jedoch ist hier der Trainingsprozess unsupervised. Außerdem arbeitet es nicht wie word2vec direkt mit dem Text, sondern mit einer Co-Occurence Matrix über welchen der Kontext der Wörter erhalten wird.

- fastText (Joulin et al., 2016)

  FastText ist im gegensatz zu word2vec kein Word-Embedding mehr, sondern ein Charachter-Embedding. Somit kann es im gegensatz zu den word2vec und GloVe Wörter vektorisieren, welche nicht im Trainingsvokabular enthalten waren.

Durch word2vec, GloVe und fastText entstehen m Vektoren, wobei m die Anzahl der Wörter in dem Satz nach dem Preprocessing ist. Anschließend können diese Vektoren unterschiedlich für die Machine-Learning Algorithmen verarbeitet werden. Um die Dimensionalität des Inputs zu reduzieren, wurde eine Aggregationsfunktion gewählt, welche einen Vektor auf jeweils einen Wert abbildet. Hier wurde das Arithmetische Mittel auf die Elemente der Vektoren gebildet. Es ist aber durchaus denkbar, die Vektoren miteinander zu konkatenieren.

2.7 Feature Engineering

Zuletzt wird das Feature Engineering betrachtet. Durch die Vektorisierung des Textes sind einige Features entstanden, jedoch kann mit zusätzlichen Features die Genauigkeit des Models verbessert werden. Hier ist das Ziel, für den Machine-Learning Algorithmus möglichst Hilfreiche Features zu erstellen. Diese Features können sowohl mit den Rohdaten als auch mit den verarbeiteten Daten erstellt werden. In dieser Arbeit werden aber die Rohdaten genommen, da diese mehr Information enthalten. Beispielsweise wurden im Preprocessing der Daten Emojis und andere Elemente gefiltert.

Folgende Features können von interesse sein:

- Anzahl der Emojis

  Aus Abbildung 12 kann abgelesen werden, dass tendenziell weibliche Teilnehmer mehr Emojis nutzen als die männlichen. Jedoch ist es schwieriger die extrovertieren Nutzer anhand dieses Features zu unterscheiden. Abbildung 13 sagt aus, dass die introvertierten Nutzer tendenziell mehr Emojis nutzen, jedoch muss hierbei beachtet werden, dass es ca. doppelt soviele introvertierte Nutzer wie extrovertierte Nutzer gibt. Somit folgt, dass die auf die Nutzeranzahlen der Klassen normalisierte Abbildung 14 wesentlich Aussagekräftiger ist.

  In Ljubešić et al. (2017) wurde der prozentuelle Anteil der Emojis im Tweet als Feature verwendet und es ist nach Tabelle 2 in Ljubešić et al. (2017) für die deutsche Sprache offensichtlich ein sehr aussagekräftiges Feature. Mit dem nächsten Feature wird das verdeutlicht.

- Bag of Word bezüglich Emojis

  Das Vorkommniss von bestimmten Emojis kann Hinweise auf die Zugehörigkeit zu einem bestimmten MBTI oder Geschlecht liefern. Somit kann beispielsweise anhand eines Emojis ausgesagt werden, dass der Autor dieser Nachricht eher weiblich als männlich ist. Durch die Abbildungen 15 und 16 wird das bildlich dargestellt. Generell ist erstmals erkennbar, dass die weiblichen Teilnehmer in erster Linie mehr Emojis in ihren Tweets verwenden (Faktor 2 bis 3), wenn nur die zehn meistgenutzten Emojis betrachtet.

- Anzahl der Hashtags, Stoppwörter, URLS und Zahlen

  Diese Elemente wurden aus dem Korpus im Preprocessing herausgenommen, da diese keine Information über das Sprachprofil des Nutzers geben. Jedoch kann die Anzahl dieser Elemente relevante Hinweise auf das Geschlecht oder den MBTI geben. Es können hier weitaus mehr Informationen aus dem Datensatz gezogen werden, wie beispielsweise die Uhrzeit an der der Tweet veröffentlicht wurde.

- Verteilung der Part-of-speech-Tags (POS-Tags)

  Welche POS-Tags ein Nutzer verwendet kann tiefergreifende Hinweise auf das linguistische Profil des Nutzers geben. Um zu verhindern, dass sich der Machine-Learning Algorithmus auf bestimmte Positionen oder sogar bestimmte Reihenfolgen der Tags konzentriert wird die Verteilung der Tags bestimmt.

- Verteilung der Dependenzen

  Die Dependenzen sind ein weiteres linguistisches Feature. Hier wird analog zu den POS-Tags vorgegangen.

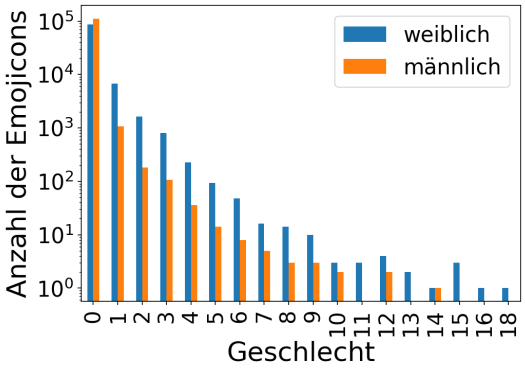

Abbildung 12: Anzahl der Emojis pro Tweet nach Geschlecht.

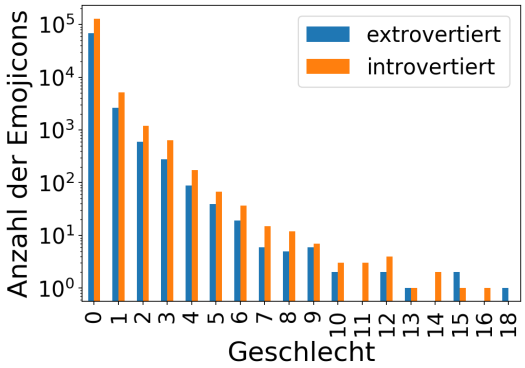

Abbildung 13: Anzahl der Emojis pro Tweeet nach dem ersten MBTI Zeichen.

2.8 Aufteilung des Datensatzes

Der Datensatz hat keine Aufteilung für den Trainingsprozess und Testprozess der Machine-Learning Algorithmen. Mit etwas meher 200 000 Datenpunkten ist der Datensatz relativ groß und wird daher in zwei Datensätze aufgeteilt mit einer Größe von

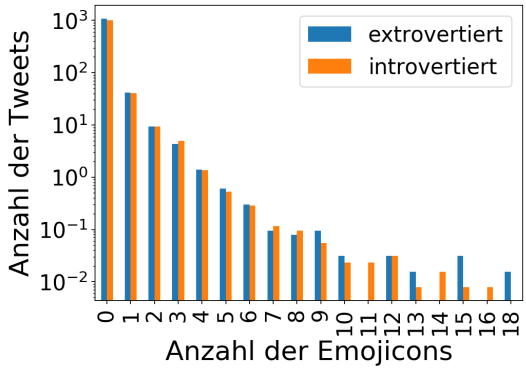

Abbildung 14: Anzahl der Tweets von jedem MBTI.

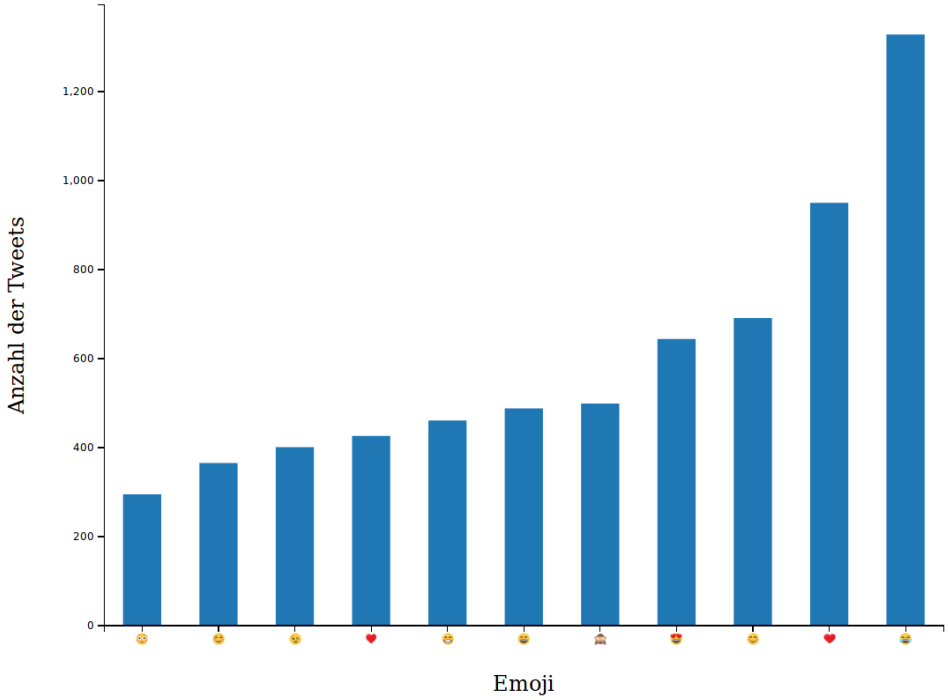

Abbildung 15: Anzahl der zehn meistgenutzten Emojis der weiblichen Teilnehmer.

100 000 Datenpunkten aufgeteilt. Einer für das Training der Algorithmen und einer für das Testen. Außerdem wird der Datensatz vor der Aufteilung mit dem Randomseed 123 gemischt.

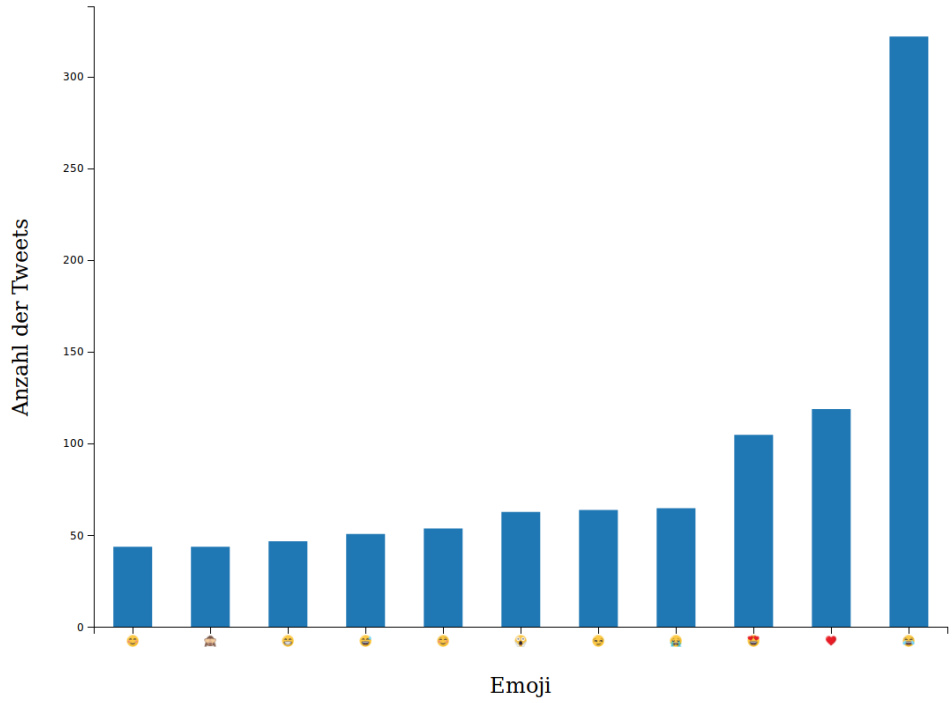

Abbildung 16: Anzahl der zehn meistgenutzten Emojis der männlichen Teilnehmer.

3 Verfahren

Nachdem auf den Datensatz eingegangen wurde, wird der Fokus nun auf die Machine-Learning Algorithmen gelegt. In diesem Kapitel wird beschrieben welche Machine-Learning Algorithmen verwendet werden, wie die entsprechenden Parametereinstellungen ausgewählt wurden um die möglichst gute Werte zu erhalten und welche Methoden genutzt werden um diese Algorithmen miteinander zu vergleichen.

3.1 Klassische Machine-Learning Algorithmen

Dieses Kapitel beschäftigt sich mit klassischen Machine-Learning Algorithmen. Hier wird erklärt welche Algorithmen verwendet werden und wie die Parameter für diese Algorithmen gewählt wurden.

3.1.1 Wahl der Algorithmen

Es werden insgesamt sechs Algorithmen aus dem klassischen Machine-Learning miteinander verglichen, dabei handelt es sich um:

- Decision Tree (L. Breiman et al., 1984)

  Die Decision Trees sind sehr einfache Klassifizierungsalgorithmen, welche einfach zu trainieren sind und dafür schon gute Ergebnisse liefern können.

- Decision Stump

  Ein Decision Stump ist ein Decision Tree welcher auf eine Höhe von eins limitiert ist, daher wird er nicht zu der Anzahl der genutzten Algorithmen gezählt. Somit hat ein Decision Stump nur ein Feature wonach er klassifizieren kann. Dieser Algorithmus wurde für den Vergleich in Betracht gezogen, da er mit Abstand der schwächste von den ausgewählten ist. Jedoch kann hiermit verdeutlich werden, wie gut Boosting funktionieren kann. Des Weiteren wird dieser Algorithmus genutzt, um einen Ausgangswert zu erhalten.

- SVM (Cortes und Vapnik, 1995)

  SVMs sind vielseitige Algorithmen aus dem klassischen Machine-Learning welche mit der vielzahl an Hyperparameter möglichst optimal an das Problem adjustiert zu werden können.

- AdaBoost (Freund und Schapire, 1999)

  Das Boosten von Algorithmen beschreibt, dass ein Algorithmus mehrfach trainiert wird und dann mittels eines Mehrheitsbeschlusses der Algorithmen die enstprechende Klasse vorhergesagt. AdaBoost geht einen Schritt weiter und passt den Datensatz in jeder Iteration an, sodass die falsch klassifizieren Datenpunkte hoffentlich richtig klassifiziert werden. Das bietet sich vorallem sehr gut bei schwachen Algorithmen an. Darüber hinaus stellt sich nach Freund und Schapire (1999) heraus, dass AdaBoost gut auf Texten abschneiden.

- Logistic Regression (Harrell, 2015, S. 311–325)

  Logistic Regression ist ein linearer Klassifierer, welcher seine nichtlinerarität durch die Simgoidfunktion erhält. Es ist erneut ein einfacher Algorithmus, welcher in Abhängigkeit der Form der Daten entsprechenden Ergebnisse liefern kann.

- Multinomial Naive Bayes (Manning et al., 2008, 208 f)

  Multinomial Naive Bayes ist ein weiterer einfacher und vorallem schnell zu trainierender Algorithmus, welcher außerdem gut auf Daten mit einer hohen Dimension funktioniert.

- Random Forest (Leo Breiman, 2001)

  Ein Random Forest ist ein Versuch das schnelle Overfitting der Decision Trees auf dem Trainingsdatensatz zu verhindern. Es werden hierbei einfach mehrere Decision Trees trainiert.

3.1.2 Wahl der Parameter

Viele der angesprochenen Algorithmen haben viele verschiedene Parameter, mit welchen diese angepasst werden konnten. Die Wahl dieser Parameter wurde mittels Gridsearch und 3-Fold Cross-validation nach einem Attribut getätigt. Hierbei wurden für die Parametersuche nur 50 000 Datenpunkte genommen, damit die Suche schneller erfolgt. Anschließend wurde das Model auf dem gesamten Testdatensatz getestet. Um die Parametereinstellungen in den einzelnen Iterationen zu vergleichen wurde als Metrik

Macro-Average-F1 genutzt, denn es kann sein, dass die Klassen ungleich verteilt wurden, somit ist die Accuracy des Algorithmus nicht aussagekräftig. Dies ist notwendig, da in Abschnitt 2.4 dargestellt wurde, dass die MBTI bzw. das Verhältnis bezüglich den introvertieren und extrovertierten Nutzer ungleichmäßig ist.

3.2 Deep-Learning

Deep Learning ist ein Gebiet des Machine-Learnings in welchem der Fokus auf neuronalen Netzen liegt. Der Vorteil von neuronalen Netzen ist, dass sie gut mit hochdimensionalen Eingaben und einer hohen Anzahl an Daten skalieren. Des Weiteren erleichtern die verschiedenen Layer der neuronalen Netze es mit den Eigenschaften der Daten umzugehen. Jedoch wird deutlich mehr Speicher und Rechenleistung benötigt. Es ist somit aufwendiger diese Algorithmen zu trainieren.

3.2.1 Gewählte Architekturen

Es werden verschiedene Architekturen betrachtet. Zuerst werden sieben Architekturen mit nur einem Inputkanal betrachtet. Hierbei handelt es sich um ein Fully Connected Neural Network, Convolutional Neural Networks, Recurrent Neural Networks und Long Short-Term Memory Neural Networks. Anschließend werden Architekturen betrachtet, in welchen zwei Inputkanale vorhanden sind (Multichannelarchitekturen). Hierbei wird der Text auf unterschiedliche Arten vektorisiert und dann dem neuronalen Netz übergeben. Beispielsweise kann der Text mit Bag of Words und fastText vektorisiert werden und anschließend dem neuronalen Netz auf diese Weise beide Darstellungen verarbeiten. Hiervon wird ein LSTM betrachet.

- Fully Connected Neural Networks (FC-NNs)

  Sehr einfache Architekturen welche keine Besonderheiten aufweisen.

- Convolutional Neural Networks (CNN)

  Die Texte können als Matrix dargestellt werden und somit vom neuronalen Netz wie ein Bild aufgefasst werden. Daher ist es interessant, ob dieser Ansatz gute Ergebnisse liefern kann.

- Recurrent Neural Networks (RNNs)
  Short-Term Memory Neural Networks (LSTMs) (Hochreiter und Schmidhuber, 1997)

  Diese speziellen neuronale Netze besitzen im Vergleich zu den vorangegangenen Netzen ein 'Erinnerungsvermögen'. Da ein Text als eine Sequenz von Werten interpretiert werden kann bietet sich diese Architekturen sehr an.

3.2.2 Wahl der Parameter

Die Parameter der neuronalen Netze wurde per Hand ausgewählt. Es hat sich herausgestellt, dass mehr als 30 Epochen die neuronalen Netze zum Overfitten bringt. Um den

entgegen zu wirken wurde Dropout (Srivastava et al., 2014) mit einer Wahrscheinlichkeit von 50% verwendet. Für das Optimieren der neuronalen Netze wurde der Adam Optimierer (Kingma und Ba, 2014) mit einer Lernrate von 0.001 genutzt.

3.3 Vom Deep-Learning zum klassischen Machine-Learning

Dieser Teil thematisiert Dimensionsreduktion mittels neuronaler Netze. Dies ist nun ein weiterer interessanter Aspekt der neuronalen Netze, denn neuronale Netze lernen Features. Somit kann nach einem Layer in den neuronalen Netzen eine passende nichtlineare Transformation der Daten stattgefunden haben, welche dem klassischen Machine-Learning Algorithmus eine höhere Genauigkeit ermöglicht. Hierfür werden die in Abschnitt 3.2 trainierten Modelle verwendet.

4 Evaluation

Dieses Kapitel befasst sich mit der Evalutaion der Algorithmen. Es werden hier jeweils der Trainingsscore und Testscore mit dem Macro-Average-F1 angegeben, wenn es nicht anders angegeben werden.

4.1 Klassischen Machine-Learning Algorithmen

Nachdem für die klassischen Machine-Learning Algorithmen mittels Gridsearch Parametereinstellungen gefunden wurden, werden diese Algorithmen nun miteinander verglichen. Zuerst werden die klassischen Machine-Learning Algorithmen veglichen. Hierbei wird der Text mit den Verfahren, welche in Abschnitt 2.4 genannt wurden, vektorisiert. Anschließend werden Features hinzugenommen, welche aus dem Feature Engineering enstanden sind. Dies wird jeweils einmal für das Geschlecht und der Persönlichkeit der Nutzer durchgeführt.

4.1.1 Auswertung ohne Feature Engineering

Zuerst werden die Ergebnisse mit dem Bag of Words Verfahren betrachtet. Es ist der Tabelle 2 zu entnehmen, dass die besten Ergebnisse durch Logistic Regression, Multinomial Naive Bayes und dem Random Forest enstanden sind. Diese Algorithmen haben einen Testscore von $> 60\%$ mit Macro-Average-F1 erreicht. Des Weiteren ist bei diesen drei Algorithmen zu beobachten, dass nur der Random Forest als einziger Algorithmus den Testdatensatz sehr gut klassifiziert hat. Dies wird durch den hohen Trainingsscore von 96.8088% deutlich. Jedoch kann hier nicht von einem Overfitting gesprochen werden, denn dieses Ensembleverfahren hat einen höheren Tesetscore als das Basisverfahren. Anschließend weist die SVM mit dem linearen Kernel etwas geringere Werte auf, nämlich 58.8095%. Vergleichsweise liefert die SVM mit dem RBF-Kernel deutlich schlechtere Ergebnisse, 55.9788% Testscore und 56.8548% sind keine besonders guten Werte. Hingegen hat der Decision Tree deutlich besser abgeschnitten. Obwohl dieser ein sehr einfacher Algorithmus ist, hat er einen Testscore von 57.2615% und einen Trainscore von 60.0843% erreicht. Letzendlich ist auch ersichtlich, dass Boosting zu keiner signifikanten Verbesserung des Decision Stumps und des Decision Trees geführt hat. Hierbei sind auf bei dem Testscore ungefähr ein Prozentpunkt mehr erreicht worden.

Anschließend stellt sich bei word2vec heraus, dass die SVMs am schlechtesten Abschneiden. Die SVM mit dem linearen und Sigmoidkernel haben einen Trainscore und Testscore von ca. 34.6%. Der RBF-Kernel hingegen ist nur minimal besser mit einem Trainscore und Testscore ca. 36%. Das Boosting auf dem Decision Tree hat zu einem Overfitting geführt, denn der Testscore ist gesunken und der Trainingsscore ist drastisch gewachsen. Mit einem Trainingsscore von ca. 99% hat dieser Algorithmus den Datensatz praktisch auswendig gelernt. Der Decision Tree hatte einen Testscore von 59.5272%, wohingegen der geboostete Decision Tree einen Testscore von ca. 54.9565% hat. Es ist weiterhin erneut zu beobachten, dass der Random Forest sehr gut den Testdatensatz gelernt hat und die besten Resultate produziert hat. Mit 61.5506% auf dem Testdatensatz ist er nur um einen sehr kleinen Anteil besser als der Decision Tree.

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	57.2615	60.0843	59.5272	69.6104
Decision Stump (DS)	53.2178	53.2027	48.7934	49.1647
SVM - linearer Kernel	58.8059	60.4004	34.6281	34.6572
SVM - RBF-Kernel	55.9788	56.8548	36.4106	36.5004
Logistic Regression	60.7730	62.2816	44.5326	54.3851
Multinomial Naive Bayes	60.9011	61.7764		
Random Forest	60.5937	96.8088	61.5506	99.9047
AdaBoost mit DT	58.6041	80.3475	54.5965	99.9006
AdaBoost mit DS	54.9445	55.0224	50.6889	51.3766
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	55.3370	59.1167	52.4959	68.8205
Decision Stump (DS)	44.1383	44.3719	49.8335	50.2961
SVM - linearer Kernel	42.5731	42.8089	34.6281	34.6571
SVM - RBF-Kernel	42.5731	42.8089	46.2973	47.0917
Logistic Regression	41.7840	42.0788	49.7377	50.1970
Multinomial Naive Bayes				
Random Forest	60.1121	99.9097	58.9380	99.9107
AdaBoost mit DT	55.2130	99.9097	54.8272	99.9097
AdaBoost mit DS	52.1185	52.8727	50.5868	51.3082

Tabelle 2: Auswertung der klassischen Machine-Learning Algorithmen mit Macro-Average-F1 bezüglich dem Geschlecht. Angaben sind in Prozent auf vier Nachkommastellen gerundet angegeben.

Folgend wird GloVe mit den Machine-Learning Algorithmen betrachtet. Analog zu word2vec ist zu erkennen, dass die SVMs die schlechtesten Resultate liefern. Erstaunlich ist, dass der lineare Kernel und der RBF-Kernel die selben Werte produziert haben. Der Testscore liegt bei 42.5731% und der Trainingsscore bei 42.8089%. Weiterhin kann zu GloVe noch abgelesen werden, dass Boosting wie bei den anderen beiden Verfahren keinen Erfolg hatte.

Zuletzt wird bezüglich dem Geschlecht fastText betrachet. Hierbei ist ersichtlich, dass im Vergleich zu den anderen Verfahren den Text zu vektorisieren kein Machine-Learning Algorithmus einen Testscore von 60% oder höher erreicht hat. Die beiden besten Algorithmen sind hier der Decision Tree und der Random Forest. Der Random Forest hat einen Testscore von 58.9380% und einen Trainingsscore von 99.9107% erzielt, wohingegen der Decision Tree 53.4959% im Testscore und 68.8205% im Trainingsscore erreicht hat. Somit ist dieser deutlich besser als die SVMs, welche einen Tesetscore und Trainingsscore zwischen ca. 34% bzw. 47% erzielt haben.

Nachdem die Machine-Learning Algorithmen bezüglich dem Geschlecht der Nutzer verglichen wurden wird selbiges bezüglich der Persönlichkeit durchgeführt.

In der Tabelle 3 ist erkennbar, dass die Eregebnisse im Verhältnis zu denen aus 2 schlech-

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	48.3883	50.6340	55.9558	65.8077
Decision Stump (DS)	39.0047	39.1379	39.0047	39.1379
SVM - linearer Kernel	46.6389	47.0248	39.0047	39.1835
SVM - RBF-Kernel	41.1118	41.2321	39.0280	39.1853
Logistic Regression	50.2638	51.1690	39.0047	39.1379
Multinomial Naive Bayes	52.4243	53.7131		
Random Forest	51.3752	96.5817	54.4560	99.8627
AdaBoost mit DT	54.9708	77.8973	51.5990	99.8452
AdaBoost mit DS	40.3860	40.5023	39.0047	39.1456
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	47.1295	39.0650	55.9558	69.6175
Decision Stump (DS)	39.0047	39.1379	39.0047	39.1379
SVM - linearer Kernel	39.0047	39.1379	39.0047	39.1379
SVM - RBF-Kernel	39.0047	39.1379	39.0047	39.1379
Logistic Regression	39.0047	39.1379	39.0047	39.1379
Multinomial Naive Bayes				
Random Forest	49.8039	99.8736	49.8039	99.8736
AdaBoost mit DT	51.4083	99.8703	51.5990	99.8453
AdaBoost mit DS	39.0047	39.1379	39.0069	39.1456

Tabelle 3: Auswertung der klassischen Machine-Learning Algorithmen mit Macro-Average-F1 bezüglich der Persönlichkeit. Angaben in Prozent gerundet auf vier Nachkommastellen.

ter sind. Es ist erkennbar, dass bei dem Bag of Words Verfahren hier der beste Classifier auf dem Testdatensatz AdaBoost mit dem Decision Tree als Basisverfahren ist. Es wurde ein Score von 54.9708% auf dem Testdatensatz und ein 77.8973% erreicht. Dies ist eine gute Möglichkeit gewesen zu demonstrieren, wie gut AdaBoost funktionieren kann. Es lässt sich weiterhin beobachten, dass der Multinomial Naie Bayes, die Logistic Regression und der Random Forest einen Testscore von 50.0% bis zu 50.2% haben, was nur um einen sehr kleinen Anteil schlechter ist.

Bei den anderen drei Methoden den Text zu vektorisieren lässt sich beobachten, dass die die selbe Informationen bezüglich der Persönlichkeit der Nutzer halten. Das ist soweit sehr interessant, denn es lässt den folgenden Schluss ziehen. Bezüglich der Persönlichkeit der Nutzer ergibt es keinen Unterschied welches Embedding gewählt wird um den Text zu vektorisieren. Hier ist aber weiterhin zu erkennen, dass Ensembleverfahren keinen Erfolg zeigen. Der Random Forest und AdaBoost Classifier overfitten.

Abschließend für diesen Teil kann folgendes Fazit gezogen werden. Verfahren wie der Decision Stump und AdaBoost mit dem Decision Stump als Basisverfahren keine nennenswerten Ergebnisse liefern. Zumal durch das Feature Engineering sehr viele Features hinzukommen und dieser Algorithmus das nicht ausnutzen kann. Daher werden diese beiden Algorithmen in den Zukünftigen Evalutation nicht mehr betrachtet. Des Weite-

ren scheint es, dass die SVM mit dem RBF-Kernel nicht so gut funktioniert wie die SVM mit dem linearen Kernel und wird daher ebenfalls nicht mehr betrachtet. Darüber hinaus scheint es, dass Bag of Words ein besseres Verfahren ist den Text zu vektorisieren als die anderen Verfahren.

4.1.2 Auswertung mit Feature Engineering

Im Anschluss werden zwei weitere Auswertungen getätigt, in welchen Feature Engineering hinzugezogen wird. In Tabelle 4 ist zu beobachten, das erneut die Algorithmen mit dem Bag of Words verfahren deutlich besser abschneiden als die mit den Embeddings. Hier hat der Logistic Regression Algorithmus den höchsten Score mit ca. 64% erreicht. Vergleichsweise hat Logistic Regression mit den Embeddings mindestens 7 Prozenpunkte schlechter auf dem Testdatensatz und Trainingsdatensatz abgeschnitten. Darüber hinaus stellt sich heraus, das Tendenziell Algorithmen besser mit dem Bag of Words Verfahren und Feature Engineering abschneiden als mit den Embeddings und dem Feature Engingeering. Die einzige Ausnahme ist der Decision Tree mit GloVe welcher ca. einen Prozentpunkt besser auf dem Testdatensatz abschneidet. Dieser Trend ist aber nur zu beobachten, wenn nach dem Geschlecht klassifiziert wird. Wenn hingegen nach der Persönlichkeit des Nutzer klassifiziert wird, so ist dieser Trend nicht mehr so stark vertreten (siehe Tabelle 5) Random Forest schneidet auf dem Embeddings deutlich besser als auf dem Bag of Words ab. Jedoch schneidet AdaBoost mit den Decision Trees als Basisverfahren und Multinomial Naive Bayes besser ab als alle andere Verfahren, wenn der Testscore in betracht gezogen wird. Besonders interessant ist, dass AdaBoost besser Funktioniert als Random Forest. Hier ist es auf dem Testdatensatz mindestens 3 Prozentpunkt besser als Random Forest.

4.2 Deep-Learning

Nun wird betrachtet wie die neuronalen Netze auf diesen Klassifizierungsproblem abschneiden. In Tabelle 6 wird nach dem Geschlecht der Nutzer klassifiziert. Es ist ein ähnliches Verhalten wie bei den klassischen Machine-Learning Algorithmen zu beobachten. Bag of Words liefert die besten Ergebnisse. In diesem Fall sowohl auf dem Testdatensatz als auch auf dem Trainingsdatensatz. Es stellt sich heraus, dass das FC-NN die Daten am besten generalisieren konnte. Weiterhin ist auffällig, dass der Testscore meistens minimal höher ist als der Trainingsscore. Selbes fällt auch in Tabelle 7 auf. Hier schneidet das FC-NN am besten auf Bag of Words ab. Weiterhin ist zu beobachten, dass bei der Klassifizierung nach der Persönlichkeit alle neuronalen Netze bei mit den Embeddings gegen die selben Werte konvergieren. Anscheinend können diese keine Darstellung liefern.

Dadurch, dass Bag of Words mit dem FC-NN am besten funktioniert wird dieses neuronale Netzt für den nächsten Schritt genutzt.

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	57.4524	60.8547	56.7480	68.9095
SVM - linearer Kernel	62.7373	64.1103	44.9366	44.9891
Logistic Regression	64.0099	65.5212	56.6434	57.0427
Multinomial Naive Bayes	62.1249	63.0343		
Random Forest	63.3526	99.9970	60.3113	60.7825
AdaBoost mit DT	61.1372	92.3312	58.1179	58.6532
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	58.6612	62.34059	56.8496	71.6991
SVM - linearer Kernel	44.8999	44.93739	44.6249	44.5452
Logistic Regression	50.6469	50.61567	56.8324	57.1178
Multinomial Naive Bayes				
Random Forest	61.2497	1.0	60.9851	1.0
AdaBoost mit DT	60.1119	1.0	60.1654	1.0

Tabelle 4: Auswertung der klassischen Machine-Learning Algorithmen zusätzlich mit Feature Engineering mit Macro-Average-F1 bezüglich dem Geschlecht. Angaben in Prozent, gerundet auf vier Nachkommastellen.

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	48.0607	51.5198	53.6482	65.2981
SVM - linearer Kernel			39.0047	39.1379
Logistic Regression	52.5464	53.8809	41.4075	41.7174
Multinomial Naive Bayes	55.6285	56.7031		
Random Forest	45.4000	99.9978	52.6476	1.0
AdaBoost mit DT	56.4467	91.0883	54.0974	1.0
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
Decision Tree (DT)	50.5713	52.6917	39.6914	39.7991
SVM - linearer Kernel	39.0047	39.1379	39.0047	39.1379
Logistic Regression	39.2024	39.3380	41.4723	41.7435
Multinomial Naive Bayes				
Random Forest	50.4096	1.0	50.3168	1.0
AdaBoost mit DT	54.0244	1.0	54.0225	1.0

Tabelle 5: Auswertung der klassischen Machine-Learning Algorithmen zusätzlich mit Feature Engineering mit Macro-Average-F1 bezüglich der Persönlichkeit. Angaben in Prozent gerundet auf vier Nachkommastellen.

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
FC-NN	50.1390	50.0601	48.1186	48.1338
CNN			34.6281	34.6281
RNN	45.8945	45.8215	44.3252	44.2818
LSTM	49.5869	50.2531	43.2192	43.0680
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
FC-NN	45.6500	45.9313	43.5364	43.9206
CNN	34.6281	34.6281	34.6281	34.6572
RNN	40.0817	40.2538	37.1176	36.9635
LSTM	42.5217	42.5615	39.6440	39.8226

Tabelle 6: Auswertung der Deep-Learning Algorithmen mit Macro-Average-F1 bezüglich dem Geschlecht. Angaben in Prozent gerundet auf vier Nachkommastellen.

	Bag of Words		word2vec	
	Testscore	Trainscore	Testscore	Trainscore
FC-NN	48.9347	49.9312	39.0047	39.1379
CNN			39.0047	39.1379
RNN	41.6047	41.7212	39.0047	39.1379
LSTM	46.9145	48.5293	39.0047	39.1379
	GloVe		fastText	
	Testscore	Trainscore	Testscore	Trainscore
FC-NN	39.1791	39.3685	39.0047	39.1379
CNN	39.1791	39.3685	39.0047	39.1379
RNN	39.0047	39.1379	39.0047	39.1379
LSTM	39.0047	39.1379	39.0047	39.1379

Tabelle 7: Auswertung der Deep-Learning Algorithmen mit Macro-Average-F1 bezüglich der Persönlichkeit. Angaben in Prozent gerundet auf vier Nachkommastellen.

	FC-NN mit Bag of Words	
	Testscore	Trainscore
Decision Tree (DT)	34.6281	34.65718
Random Forest	44.2437	43.12971

Tabelle 8: Auswertung der klassischen Machine-Learning Algorithmen mit Macro-Average-F1 bezüglich dem Geschlecht. Angaben in Prozent gerundet auf vier Nachkommastellen.

	FC-NN mit Bag of Words	
	Testscore	Trainscore
Decision Tree (DT)	41.2033	40.7486
Random Forest	47.1928	51.8388

Tabelle 9: Auswertung der klassischen Machine-Learning Algorithmen mit Macro-Average-F1 bezüglich der Persönlichkeit. Angaben in Prozent gerundet auf vier Nachkommastellen.

4.3 Vom Deep-Learning zum klassischen Machine-Learning

Zuletzt wird die Wechselwirkung von neuronalen Netzten zu klassischen Machine-Learning Algorithmen beobachtet. Hier wird beobachtet, ob das neuronale Netz dem klassischen Machine-Learning Algorithmus in diesem Klassifizierungsproblem eine bessere Darstellung der Daten bieten kann.

Aus den Tabellen 8 und 9 ist ersichtlich, dass diese Darstellung einen negativen Einfluss auf die beiden Machine-Learning Algorithmen hat. Der Grund dafür ist die ohnehin nicht optimale Leistung des neuronalen Netzes.

5 Abschluss

5.1 Fazit

Nachdem die Machine-Learning Verfahren evaluiert wurden, werden nun diese kritisch begutachtet. Wenn nur die klassischen Machine-Learning Algorithmen betrachetet werden dann ergibt sich, dass das Bag of Words Verfahren sich am besten für Author Profiling mit Multinomial Naive Bayes, Logistic Regression und Random Forest eignet. Die die Embeddings liefern hingegen nur mit dem Random Forest vergleichbare Ergebnisse. Dies wird ebenfalls deutlich, sobald Feature Engineering betrieben wird. Hier ergibt sich erneut das selbe Bild.

Die neuronalen Netze haben im Vergleich deutlich schlechter abgeschnitten und haben Probleme gehabt überhaupt an den Trainingsdatensatz zu fitten ohne dabei zu overfitten. Hierbei kann das Problem an dem Datensatz liegen, da das selbe Problem bei einigen anderen klassischen Machine-Learning Algorithmen zu beobachten ist.

Weiterhin konnte auf diesem Datensatz kein Erfolg mit der Dimensionsreduktion durch die neuronalen Netze erzielt werden. Es ist wahrscheinlich, dass das neuronale Netzt die Daten nicht gut genug abstrahieren konnte.

5.2 Zukünftige Arbeiten

Aus Zeitgründen konnten einige interessante Ansätze nicht untersucht werden andererseits sind weiter Ideen während der Bearbeitung dieser Arbeit entstanden, welche vom dem Thema dieser Arbeit abweichen.

Es wurden in dieser Arbeit bisher keine komplexeren Deep-Learning Ansätze betrachtet, das kann in Zukünftigen Arbeiten umgesetzt werden. Es kann beispielsweise Multitasklearning verwendet werden um nach mehreren Merkmalen gleichzeitig zu klassifizieren, da die Informationsdichte hier deutlich besser genutzt werden kann.

Ein weiterer Aspekt der hier nicht aufgegriffen wurde ist Unsupervised Machine-Learning. Nicht immer sind gelabelte Daten vorhanden, somit kann beobachtet werden ob es damit möglich ist Author Profiling zu betreiben, wenn keine Daten gelabelt sind und welche Merkmale die Algorithmen finden.

Um im Bereich der Text zu bleiben; es wurde das Problem mit Datensätzen von Twitter kritisiert, dass diese nicht permanent sind. Eine weitere Möglichkeit wäre beispielsweis es ein Generative Adversarial Neural Network (GAN) (Goodfellow et al., 2018) zu trainieren, welches legitime Texte generiert. Hiermit können dann ggf. bestimmte Merkmale der Autoren von neuronalen Netzen besser nachvollzogen werden, wenn diese generiert werden können. Desweiteren würde ein Datensatz generiert werden, welche folglich permanent sind.

Weiterhin wurde in dieser Arbeit bisher nur der Text und einige Metadaten welche durch den Text erfahren wurden in Betracht gezogen. In Ljubešić et al. (2017) wurde sehr viel Feature Engineering betrieben, beispielsweise wurde die Uhrzeit der Veröffentlichung der Textnachricht verwendet. Jedoch wurden die Bilder der Nutzer nicht in Betracht ge-

zogen. Ein weiteres interessantes Thema ist daher, ob die Bilder die von den Nutzern hochgeladen wurden Hinweise auf die Merkmale der Autoren geben. Es kann somit Author Profiling auf Bildern untersucht werden.

Zuletzt können Datensätze welche auf verschiedenen Plattformen gesammelt wurden untersucht werden. Hierbei wird ein Machine-Learning Algorithmus auf einem Datensatz von einer Plattform wie Twitter trainiert werden und anschließend untersucht werden wie gut dieser Algorithmus auf einem Datensatz von beispielsweise facebook.com oder reddit.com klassifiziert.

References

L. Breiman, J. Friedman, C.J. Stone und R.A. Olshen (1984). *Classification and Regression Trees*. Taylor & Francis.

Leo Breiman (Okt. 2001). „Random Forests“. In: *Machine Learning* 45.1, S. 5–32.

Corinna Cortes und Vladimir Vapnik (1995). „Support-vector networks“. In: *Machine Learning*.

The Myers & Briggs Foundation (2018). *The Myers-Brigss Type Indicator - MBTI Basics*. Zuletzt zugegriffen am 09-08-2018.

Yoav Freund und Robert E. Schapire (1999). „A Short Introduction to Boosting“. In: *Journal of Japanese Society for Artifical Intelligence*.

Ian J. Goodfellow, Patrick D. McDaniel und Nicolas Papernot (2018). „Making machine learning robust against adversarial inputs“. In: *Commun. ACM* 61.7, S. 56–66.

Frank E. Harrell (2015). „Ordinal Logistic Regression“. In: *Regression Modeling Strategies: With Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis*. Springer International Publishing, S. 311–325.

Sepp Hochreiter und Jürgen Schmidhuber (1997). „Long Short-Term Memory“. In: *Neural Computation* 9.8, S. 1735–1780.

Armand Joulin, Edouard Grave, Piotr Bojanowski und Tomas Mikolov (2016). „Bag of Tricks for Efficient Text Classification“. In: *CoRR*. arXiv: 1607.01759.

Diederik P. Kingma und Jimmy Ba (2014). „Adam: A Method for Stochastic Optimization“. In: *CoRR* abs/1412.6980. arXiv: 1412.6980.

Long-Yi Lin (2010). „The relationship of consumer personality trait, brand personality and brand loyalty: An empirical study of toys and video games buyers“. In: *Journal of Product & Brand Management*.

Nikola Ljubešić, Darja Fišer und Tomaž Erjavec (2017). „Language-independent Gender Prediction on Twitter“. In: *Proceedings of the Second Workshop on NLP and Computational Social Science*. Association for Computational Linguistics, S. 1–6.

Christopher D Manning, Prabhakar Raghavan und Hinrich Schütze (2008). *Introduction to Information Retrieval*. Bd. 39. Cambridge University Press.

Tomas Mikolov, Kai Chen, Greg Corrado und Jeffrey Dean (2013). „Efficient Estimation of Word Representations in Vector Space“. In: *CoRR*. arXiv: 1301.3781.

Mishra, Del Tredici, Yannakoudakis und Shutova (2018). „Author Profiling for Abuse Detection“. In: *COLING*.

Francisco M. Rangel Pardo, Paolo Rosso, Moshe Koppel, Efstathios Stamatatos und Giacomo Inches (2013). „Overview of the Author Profiling Task at PAN 2013“. In: *Working Notes for CLEF 2013 Conference*.

Jeffrey Pennington, Richard Socher und Christopher D. Manning (2014). „GloVe: Global Vectors for Word Representation“. In: *Empirical Methods in Natural Language Processing (EMNLP)*, S. 1532–1543.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever und Ruslan Salakhutdinov (2014). „Dropout: A Simple Way to Prevent Neural Networks from Overfitting“. In: *Journal of Machine Learning Research*.

Inc. Twitter (2018). *Developer Agreement and Policy*. Zuletzt zugegriffen: 13.08.2018.

Ben Verhoeven, Walter Daelemans und Barbara Plank (Mai 2016). „TwiSty: a multilingual Twitter Stylometry corpus for gender and personality profiling“. In: *Proceedings of*

the 10th Annual Conference on Language Resources and Evaluation (LREC 2016). ELRA. ELRA.

David H. Wolpert und William G. Macready (1995). „No free lunch theorems for optimization“. In: *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*.

Abbildungsverzeichnis

Tabellenverzeichnis