Projekt: NLP DLBAIPNLP01_D

Projektbericht

Studiengang: Angewandte Künstliche Intelligenz

Sven Behrens

Matrikelnummer: 42303511

Tutor: Prof. Dr. Maja Popovic

15. Dezember 2025

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API Application Programming Interface

1 Einleitung

Durch die zunehmende Digitalisierung werden Benutzerbewertungen sowohl zur Beurteilung von Produkten als auch in sozialen Medien immer wichtiger, sowohl für Kunden, um einen schnellen Überblick zu bekommen, als auch für Unternehmen, um Feedback systematisch auszuwerten. So werden täglich Millionen von Nutzermeinungen erzeugt, deren manuelle Analyse längst nicht mehr praktikabel ist. Die automatisierte Sentimentanalyse von Produktrezensionen stellt dabei eine zentrale Herausforderung im Bereich des Natural Language Processing dar, deren Bewältigung für Unternehmen maßgeblich zur Produktverbesserung und Kundenzufriedenheitsanalyse beiträgt. Vor diesem Hintergrund wurde im Rahmen des Moduls „Projekt: NLPän der IU Internationalen Hochschule ein umfassendes System zur Sentimentanalyse von Amazon-Produktbewertungen entwickelt, das sowohl klassische Machine-Learning-Verfahren als auch moderne Transformer-Architekturen systematisch vergleicht.

Das primäre Projektziel bestand in der Entwicklung und dem Vergleich verschiedener Klassifikationsansätze zur automatischen Sentiment-Erkennung aus englischsprachigen Produktrezensionen. Die zentrale Forschungsfrage konzentrierte sich darauf, wie sich traditionelle Machine-Learning-Verfahren, mit Modellen wie Logistic Regression gegenüber modernen vortrainierten Sprachmodellen wie DistilBERT hinsichtlich Klassifikationsgenauigkeit und Praxistauglichkeit verhalten.

Die Datenbasis bildeten Amazon-Produktrezensionen aus drei Kategorien, Automotive, Pet Supplies und Video Games, mit insgesamt 150.000 Datensätzen, die stratifiziert in Trainings- (70%), Validierungs- (15%) und Testdaten (15%) aufgeteilt wurden. Jede Rezension umfasst Titel, Text und eine Sternebewertung von 1 bis 5, wobei sowohl eine 5-Klassen-Klassifikation als auch eine aggregierte 3-Klassen-Sentimentanalyse (negativ, neutral, positiv) untersucht wurden.

Die methodische Vorgehensweise gliederte sich in mehrere aufeinander aufbauende Phasen. Nach einer initialen Datenaufbereitung mit Textbereinigung, Lemmatisierung und Stopwort-Entfernung erfolgte zunächst die Implementierung klassischer Modelle unter Verwendung von TF-IDF-Vektorisierung mit bis zu 20.000 Features. Hierbei wurden Logistic Regression, Naive Bayes, SVM, Random Forest und Gradient Boosting systematisch evaluiert. Parallel dazu wurde ein Fine-Tuning des DistilBERT-Modells auf dem vollständigen Trainingsdatensatz durchgeführt sowie ein vortrainiertes BERT-Modell für Sentimentanalyse im Zero-Shot-Modus getestet.

Der vorliegende Projektbericht dokumentiert systematisch den gesamten Entwicklungsprozess. Nach dieser Einleitung folgt die detaillierte Beschreibung der Methodik, einschließlich der Datenaufbereitung sowie der Modellauswahl und -konfiguration. Anschließend werden die Trainingsabläufe und deren Ergebnisse ausführlich dargestellt, wobei das beste Modell (DistilBERT) eine Testgenauigkeit von 80,91% erreichte, während die klassische Logistic Regression mit 76,66% konkurrenzfähige Referenzleistung erzielte. Abschließend werden die gewonnenen Erkenntnisse in einem Fazit zusammengefasst und kritisch reflektiert.

2 Hauptteil

2.1 Hardwareauswahl

Zu Projektbeginn standen drei Alternativen zur Verfügung: ein MacBook Pro mit Apple-M2-Chip, ein Windows-PC mit NVIDIA RTX 3060 sowie verschiedene Cloud-Computing-Anbieter.

Für das Training der klassischen Machine-Learning-Modelle wie Logistic Regression, Naive Bayes und SVM war sowohl das MacBook als auch der Windows-PC ausreichend, da diese Verfahren primär CPU-basiert arbeiten. Für das Fine-Tuning der BERT-Modelle schied das MacBook jedoch aufgrund der fehlenden CUDA-Unterstützung aus.

Das Training wurde daher auf dem Windows-11-Computer (Intel i7-12700K, 32 GB DDR5, NVIDIA RTX 3060 mit 12 GB VRAM) durchgeführt. Da selbst das längste Training (DistilBERT auf 105.000 Samples) mit etwa zwei Tagen in einem akzeptablen Rahmen blieb, waren zusätzliche Cloud-Lösungen nicht erforderlich.

2.2 Projektumgebung einrichten

Zu Beginn des Projekts wurde ein GitHub-Repository angelegt. Obwohl nur eine Person am Projekt arbeitet, ermöglicht GitHub eine nachvollziehbare Versionsverwaltung und ein einfaches Zurücksetzen auf frühere Stände. Anschließend wurde eine grundlegende Verzeichnisstruktur erstellt und mit `venv` eine virtuelle Umgebung eingerichtet, in der zentrale Bibliotheken installiert wurden.

2.3 Datenbeschaffung

Als Datenquelle diente der Amazon Reviews 2023 Datensatz Hou et al., 2024, der über 500 Millionen Produktbewertungen aus verschiedenen Kategorien umfasst und für akademische Forschungszwecke frei verfügbar ist.

Bei der Auswahl der Kategorien wurden zunächst kleinere Kategorien wie Gift Cards in Betracht gezogen. Diese erwiesen sich jedoch als ungeeignet, da die zugehörigen Rezensionen häufig nur aus sehr kurzen Texten wie „gutöder „schnelle Lieferung"bestanden, die für eine aussagekräftige Sentimentanalyse nicht ausreichend sind.

Stattdessen wurden die Kategorien Automotive, Pet Supplies und Video Games ausgewählt. Diese Kategorien zeichnen sich durch ausführlichere Rezensionen aus, in denen Nutzer ihre Erfahrungen detailliert beschreiben. Zudem repräsentieren sie unterschiedliche Produkttypen, wodurch die Generalisierungsfähigkeit der trainierten Modelle besser evaluiert werden kann.

Aus jeder Kategorie wurden jeweils 50.000 Rezensionen entnommen, sodass ein balancierter Gesamtda-

tensatz von 150.000 Datensätzen entstand.

2.4 Datenanalyse

2.4.1 Datensatzstruktur

In einem ersten Schritt wurden die Rohdaten untersucht und ihre Struktur analysiert. Die Rezensionen liegen im JSONL-Format vor, wobei jede Zeile einen JSON-Datensatz repräsentiert. Table 1 zeigt die verfügbaren Felder und ihre Bedeutung.

Tabelle 1: Struktur der Amazon-Rezensionsdaten im JSONL-Format

Feld	Beschreibung	Beispiel
rating	Sternebewertung (1.0–5.0)	5.0
title	Titel der Rezension	„Great product!"
text	Ausführlicher Rezensionstext	„Item came as described..."
images	Angehängte Bilder	[]
asin	Amazon Produkt-ID	B01LZA8SGZ
parent_asin	Übergeordnete Produkt-ID	B0BV88374L
user_id	Anonymisierte Benutzer-ID	AGXVBIUFLFGMV...
timestamp	Unix-Zeitstempel	1513092936205
helpful_vote	Anzahl hilfreicher Stimmen	0
verified_purchase	Verifizierter Kauf	true

Die Felder `title` und `text` wurden zu einem zusammenhängenden Eingabetext (X) kombiniert, während das Feld `rating` als Zielklasse (y) mit Werten von 1 bis 5 Sternen dient. Zusätzlich wurde die Produktkategorie gespeichert, um kategoriespezifische Evaluationen zu ermöglichen.

Anschließend erfolgte die Aufteilung der Daten in 70% Trainings-, 15% Validierungs- und 15% Testdaten.

2.5 Vorverarbeitungs-Pipeline

Die folgenden Ausführungen zur Textvorverarbeitung und Feature-Extraktion orientieren sich an den Grundlagen aus Schaaff (2025, S. 62–78).

Die Textvorverarbeitung bildet einen entscheidenden Schritt zur Vorbereitung der Rohtexte für die maschinelle Verarbeitung. Ziel ist es, Rauschen zu reduzieren und die relevanten sprachlichen Merkmale zu extrahieren.

2.5.1 Textbereinigung und Normalisierung

Die implementierte Pipeline verarbeitet jeden Rezensionstext in mehreren aufeinander aufbauenden Schritten:

1. **Textkombination**: Zusammenführung von `title` und `text` zu einem Eingabestring
2. **Kleinschreibung**: Konvertierung aller Zeichen zu Kleinbuchstaben zur Vereinheitlichung
3. **HTML-Bereinigung**: Entfernung von HTML-Tags mittels regulärer Ausdrücke
4. **URL-Entfernung**: Filterung von Weblinks, die keine semantische Relevanz besitzen
5. **Zahlenentfernung**: Eliminierung numerischer Werte
6. **Satzzeichenentfernung**: Entfernung aller Interpunktionszeichen
7. **Tokenisierung**: Zerlegung des Textes in einzelne Wörter mittels NLTK
8. **Stoppwort-Filterung**: Entfernung häufiger englischer Funktionswörter (the, is, at, ...)
9. **Lemmatisierung**: Rückführung der Wörter auf ihre Grundform mittels WordNetLemmatizer
10. **Längenfilterung**: Entfernung von Tokens mit weniger als 2 Zeichen

2.5.2 Feature-Extraktion

Für die Transformation der vorverarbeiteten Texte in numerische Vektoren stehen verschiedene Methoden zur Verfügung. Table 2 vergleicht die gängigsten Ansätze.

Tabelle 2: Vergleich der Feature-Extraktionsmethoden

Methode	Vorteile	Nachteile
Bag of Words	Einfach, schnell, interpretierbar	Ignoriert Worthäufigkeit im Korpus, große sparse Matrizen
TF-IDF	Gewichtet wichtige Wörter höher, reduziert Rauschen, bewährt für Klassifikation	Ignoriert Wortkontext und -reihenfolge
Word2Vec / GloVe	Erfasst semantische Ähnlichkeit, dichte Vektoren	Benötigt viele Trainingsdaten, schwerer interpretierbar

Für die klassischen Machine-Learning-Modelle wurde **TF-IDF** (Term Frequency-Inverse Document Frequency) gewählt. Diese Entscheidung basiert auf mehreren Faktoren: TF-IDF ist für Sentiment-Klassifikation bewährt, ermöglicht schnelles Training ohne GPU und bietet Interpretierbarkeit, es lässt sich nachvollziehen, welche Wörter zur Klassifikation beitragen.

Tabelle 3: Konfiguration der TF-IDF-Vektorisierung

Parameter	Wert	Beschreibung
max_features	10.000	Maximale Anzahl der Features
ngram_range	(1, 2)	Uni- und Bigramme
min_df	2	Term muss in mind. 2 Dokumenten vorkommen
max_df	0,95	Term darf in max. 95% der Dokumente vorkommen

Die Verwendung von Bigrammen ermöglicht die Erfassung von Wortpaaren wie „not good” oder „very bad”, die für die Sentimentanalyse besonders relevant sind, da sie Negationen und Verstärkungen berücksichtigen.

2.6 Modellauswahl

Um einen fundierten Vergleich zwischen klassische Machine-Learning-Verfahren und Transformer-basierte Sprachmodelle zu ermöglichen, wurden folgende Modelle evaluiert.

2.6.1 Klassische Machine-Learning-Modelle

Als Baseline dienten etablierte Klassifikationsverfahren in Kombination mit TF-IDF-Vektorisierung. Die Term Frequency-Inverse Document Frequency (TF-IDF) transformiert Texte in numerische Vektoren, wobei häufig vorkommende, aber wenig aussagekräftige Wörter herabgewichtet werden.

Folgende Modelle wurden implementiert und evaluiert:

- **Logistic Regression**: Bei der logistischen Regression wird eine S-förmige logistische Funktion (Sigmoidfunktion) an die Daten gefittet, wobei der Ausgabewert die Wahrscheinlichkeit einer bestimmten Klassenzugehörigkeit ausdrückt (IU Internationale Hochschule, 2025).
- **Naive Bayes**: Dieser Klassifikator beruht auf dem Satz von Bayes und geht von der Annahme aus, dass die Verteilungen der berücksichtigten Merkmale voneinander unabhängig sind (IU Internationale Hochschule, 2025).
- **Support Vector Machine (LinearSVC)**: Das Verfahren basiert auf der Einpassung einer Klassifizierungsgrenze (Hyperebene) in den hochdimensionalen Feature-Raum, wobei neue Beobachtungen je nach Position relativ zur Hyperebene klassifiziert werden (IU Internationale Hochschule, 2025).
- **Random Forest**: Diese Ensemble-Methode bündelt einzelne Entscheidungsbäume durch Bagging zu einem starken Schätzer, dessen Vorhersage durch Aggregation der Einzelvorhersagen ermittelt wird (IU Internationale Hochschule, 2025).
- **Gradient Boosting**: Im Gegensatz zu Random Forest erfolgt das Training der Entscheidungsbäume sequenziell, wobei jeder Schätzer die Fehler des vorherigen korrigiert (IU Internationale Hochschule, 2025).

Die TF-IDF-Vektorisierung wurde mit bis zu 20.000 Features und Uni- sowie Bigrammen konfiguriert. Zusätzlich wurde eine Textvorverarbeitung mit Kleinschreibung, Entfernung von HTML-Tags, URLs und Sonderzeichen sowie Lemmatisierung durchgeführt.

2.6.2 Transformer-basierte Modelle

Für den Vergleich mit modernen Deep-Learning-Ansätzen wurden zwei BERT-basierte Modelle ausgewählt:

DistilBERT Sanh et al., 2019 ist eine komprimierte Version des ursprünglichen BERT-Modells Devlin et al., 2018, die durch Knowledge Distillation erzeugt wurde. Mit 66 Millionen Parametern ist DistilBERT etwa 40% kleiner als BERT, behält jedoch 97% der Sprachverständnisfähigkeiten bei. Diese Eigenschaften machen DistilBERT besonders geeignet für Projekte mit begrenzten Hardwareressourcen.

Als zweiter Ansatz wurde ein **vortrainiertes BERT-Modell für Sentimentanalyse** (nlptown/bert-base-multilingual-uncased-sentiment) im Zero-Shot-Modus getestet. Dieses Modell wurde bereits auf Produktbewertungen trainiert und ermöglicht eine Klassifikation ohne zusätzliches Fine-Tuning auf dem eigenen Datensatz.

2.7 Training

2.7.1 Erstes Training der klassischen Modelle

Das initiale Training erfolgte mit drei klassischen Modellen: Logistic Regression, Multinomial Naive Bayes und LinearSVC. Das Modell mit der höchsten Validierungsgenauigkeit wurde automatisch ausgewählt und auf dem Testdatensatz evaluiert.

Die Evaluation umfasste Accuracy, Classification Report und Confusion Matrix. Das beste Modell wurde abschließend gespeichert.

Die initiale Evaluation zeigte eine deutliche Asymmetrie in der Klassifikationsleistung. Während 5-Sterne-Bewertungen mit einem Recall von 97% und einer Precision von 78% zuverlässig erkannt wurden, offenbarten die mittleren Klassen erhebliche Schwächen.

Die Confusion Matrix (siehe Table 4 im Anhang) verdeutlicht das Kernproblem: Das Modell tendierte dazu, Rezensionen aller Klassen als 5-Sterne-Bewertungen zu klassifizieren. So wurden beispielsweise 163 der 1-Stern-Bewertungen fälschlicherweise als 5 Sterne eingestuft, bei 4-Stern-Bewertungen waren es sogar 381 von 528.

2.7.2 Aufbau der Experiment-Pipeline

Die Erkenntnisse aus dem ersten Training zeigten, dass systematische Experimente mit verschiedenen Konfigurationen erforderlich sind. Um eine effiziente Durchführung und Vergleichbarkeit zu gewährleisten, wurde eine modulare Pipeline-Architektur implementiert, die aus drei aufeinander aufbauenden Komponenten besteht.

Die erste Komponente bildet das Preprocessing-Modul, das die Rohtexte für die maschinelle Verarbeitung vorbereitet. Folgende Verarbeitungsschritte können dabei flexibel aktiviert oder deaktiviert werden: Konvertierung zu Kleinbuchstaben, Entfernung von HTML-Tags, URLs, Zahlen und Satzzeichen sowie die Filterung englischer Stoppwörter. Für die Wortnormalisierung stehen zwei Alternativen zur Verfügung: der Porter Stemmer, der Wörter auf ihren Stamm reduziert, sowie der WordNet Lemmatizer, der Wörter auf ihre Grundform zurückführt. Zusätzlich kann eine minimale Token-Länge definiert werden, um sehr kurze Wortfragmente auszuschließen.

Die zweite Komponente umfasst die Feature-Extraktion, die den vorverarbeiteten Text in numerische Vektoren transformiert. Hierbei kann zwischen TF-IDF-Vektorisierung und einfacher Count-Vektorisierung (Bag-of-Words) gewählt werden. Die Anzahl der Features ist zwischen 5.000 und 20.000 konfigurierbar. Durch die Einstellung der N-Gramm-Range lassen sich ausschließlich Unigramme, zusätzlich Bigramme oder auch Trigramme berücksichtigen. Weitere Parameter wie minimale und maximale Dokumentfrequenz, sublineare TF-Skalierung sowie die IDF-Gewichtung ermöglichen eine Feinabstimmung der Feature-Repräsentation.

Die dritte Komponente stellt das Training-Modul dar, das fünf verschiedene Klassifikationsverfahren unterstützt: Logistic Regression, Multinomial Naive Bayes, Linear Support Vector Machine, Random Forest und Gradient Boosting. Für jedes Modell sind sinnvolle Standardparameter hinterlegt, die bei Bedarf überschrieben werden können.

Die gesamte Konfiguration erfolgt über YAML-Dateien, wodurch Experimente reproduzierbar dokumentiert werden. Jedes Experiment erhält automatisch einen Zeitstempel und speichert alle Ergebnisse strukturiert ab, einschließlich der verwendeten Konfiguration, aller Metriken, der Confusion Matrix sowie des trainierten Modells. Ein zusätzliches Vergleichsskript ermöglicht die tabellarische Gegenüberstellung aller durchgeführten Experimente.

2.7.3 Erstes Training mit 30.000 Datensätzen

Mit der implementierten Pipeline wurde ein erstes Training auf einem reduzierten Datensatz mit 30.000 Rezensionen durchgeführt. Die detaillierten Ergebnisse aller sieben Experimente sind in Table 5 im Anhang aufgeführt. Die Evaluation zeigt, dass Logistic Regression in Kombination mit Lemmatisierung und Bigrammen die höchste Accuracy von 74,29% erreichte. Bemerkenswert ist jedoch, dass Naive Bayes und LinearSVC trotz geringerer Accuracy bessere F1-Macro-Werte erzielten, was auf eine ausgewogenere Klassifikation über alle fünf Klassen hindeutet. Die Confusion Matrizen im Anhang (Tables 6 to 8) verdeutlichen diese Unterschiede: Während Random Forest nahezu alle Rezensionen als 5 Sterne klassifiziert, zeigt Naive Bayes eine breitere Verteilung über alle Klassen.

Das Hauptproblem aller Modelle bleibt die unbalancierte Klassenverteilung: Mit 64% 5-Sterne-Bewertungen

im Datensatz erreichen alle Modelle hohe Recall-Werte für die Mehrheitsklasse (bis zu 97%), während die mittleren Klassen (2, 3, 4 Sterne) mit Recall-Werten unter 35% stark unterrepräsentiert bleiben. Die Kategorie Automotive zeigte durchgängig die besten Ergebnisse, was auf eindeutigere Sentiment-Signale in dieser Produktkategorie hindeutet.

2.7.4 Training mit 150.000 Datensätzen

Nach der initialen Evaluation wurde das Training auf den vollständigen Datensatz mit 150.000 Rezensionen (105.000 Training, 22.500 Validierung, 22.500 Test) ausgeweitet. Die detaillierten Ergebnisse sind in Table 9 im Anhang aufgeführt.

Die Skalierung auf einen 150.000 Datensatz führte zu einer deutlichen Verbesserung der Klassifikationsleistung. Logistic Regression mit 20.000 Features erreichte die höchste Accuracy von 76,77%, gefolgt von der Baseline-Konfiguration mit 76,66%. Der F1-Macro-Score verbesserte sich von durchschnittlich 0,44 auf etwa 0,51, was auf eine bessere Erkennung der unterrepräsentierten Klassen hindeutet.

Table 10 im Anhang zeigt den direkten Vergleich zwischen beiden Datensatzgrößen. Die Verbesserungen betragen je nach Modell zwischen 0,58 und 2,55 Prozentpunkten, wobei Logistic Regression am stärksten profitierte. Naive Bayes zeigte mit nur 0,58 Prozentpunkten Verbesserung die geringste Skalierungseffizienz, was auf die vereinfachenden Annahmen des Modells zurückzuführen ist.

Das Klassenungleichgewicht bleibt weiterhin das zentrale Problem: Die mittleren Klassen (2, 3, 4 Sterne) erreichen auch mit mehr Trainingsdaten Recall-Werte von maximal 37%, während die 5-Sterne-Klasse konstant bei über 96% liegt. Die Kategorie Automotive erzielte mit etwa 78% durchgängig die besten Ergebnisse.

2.7.5 Anpassung der Klassengewichtung

Um das Klassenungleichgewicht direkt zu adressieren, wurde die `class_weight='balanced'`-Option von Scikit-learn evaluiert (Pedregosa et al., 2011). Diese Methode gewichtet Trainingsbeispiele inversiv proportional zu ihrer Klassenhäufigkeit.

Für den vorliegenden Datensatz resultieren daraus Gewichtungen von etwa 0,31 für die dominante 5-Sterne-Klasse bis zu 3,92 für die seltene 2-Sterne-Klasse. Fehler bei unterrepräsentierten Klassen werden somit bis zu zwölfmal stärker bestraft als bei der Mehrheitsklasse.

Die detaillierten Ergebnisse sind in Table 14 im Anhang aufgeführt. Die Klassengewichtung führte wie erwartet zu einer deutlichen Verbesserung der Recall-Werte für die mittleren Klassen: Rating 2 stieg von 17% auf 39%, Rating 3 von 25% auf 40% und Rating 4 von 26% auf 48%. Gleichzeitig sank jedoch der Recall der 5-Sterne-Klasse von 96% auf 75%.

Diese Verschiebung resultierte in einer reduzierten Gesamtaccuracy: Logistic Regression fiel von 76,66% auf 66,85%, da die 5-Sterne-Klasse 64% der Testdaten ausmacht. Der F1-Macro-Score blieb hingegen nahezu konstant (50,73% vs. 50,36%), was die ausgeglichenere Klassifikation widerspiegelt.

Die Wahl zwischen beiden Ansätzen hängt vom Anwendungsfall ab: Für maximale Gesamtgenauigkeit eignet sich das Standardmodell, für eine ausgeglichene Erkennung aller Sentiment-Stufen die gewichtete Variante.

3 Fazit

3.1 Zielerreichung und Projektergebnisse

3.2 Kritische Reflexion und gewonnene Erkenntnisse

3.3 Verbesserungspotenziale und Optimierungsansätze

3.4 Ausblick

Literatur

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Hou, Y., Li, J., He, Z., Yan, A., Chen, X., & McAuley, J. (2024). Bridging Language and Items for Retrieval and Recommendation. *arXiv preprint arXiv:2403.03952*.

IU Internationale Hochschule. (2025). *Maschinelles Lernen – Supervised Learning* [Kurscode: DLBDSMLSL01_D].

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Schaaff, K. (2025). *Einführung in Natural Language Processing* [Kurscode: DLBAIINLP01_D]. IU Internationale Hochschule GmbH.

Verzeichnis der Anhänge

Anhang

A: Detaillierte Ergebnisse der Trainings

Tabelle 4: Confusion Matrix des ersten Trainings

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	370	3	10	10	163
2 Sterne	82	24	23	19	82
3 Sterne	55	8	49	34	152
4 Sterne	25	2	15	105	381
5 Sterne	37	2	12	41	2796

Tabelle 5: Vergleich der Pipeline-Experimente (30.000 Datensätze)

Experiment	Konfiguration	Accuracy	F1 Macro
baseline	LogReg, Lemma, Bigrams, 10k	74,29%	0,443
stemming_comparison	LogReg, Stemming, Bigrams	74,27%	0,442
high_features	LogReg, Lemma, Trigrams, 20k	74,22%	0,442
unigrams_only	LogReg, Lemma, Unigrams	73,98%	0,439
svm_linear	LinearSVC, Lemma, Bigrams	73,60%	0,474
naive_bayes	MultinomialNB, Count-Vec	72,18%	0,476
random_forest	RandomForest, 5k Features	70,04%	0,337

Tabelle 6: Confusion Matrix – Logistic Regression (Baseline, 30k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	368	4	10	10	164
2 Sterne	81	23	23	18	85
3 Sterne	55	8	48	34	153
4 Sterne	25	2	16	106	379
5 Sterne	36	3	10	41	2798

Tabelle 7: Confusion Matrix – Naive Bayes (30k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	397	33	33	21	72
2 Sterne	86	46	37	22	39
3 Sterne	58	26	67	58	89
4 Sterne	31	11	50	172	264
5 Sterne	70	21	50	181	2566

Tabelle 8: Confusion Matrix – Random Forest (30k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	200	0	0	0	356
2 Sterne	25	13	0	0	192
3 Sterne	15	0	18	0	265
4 Sterne	9	0	0	46	473
5 Sterne	10	0	0	3	2875

Tabelle 9: Vergleich der Pipeline-Experimente (150.000 Datensätze)

Experiment	Konfiguration	Accuracy	F1 Macro
high_features	LogReg, Lemma, Trigrams, 20k	76,77%	0,508
baseline	LogReg, Lemma, Bigrams, 10k	76,66%	0,507
stemming_comparison	LogReg, Stemming, Bigrams	76,61%	0,507
svm_linear	LinearSVC, Lemma, Bigrams	76,12%	0,502
unigrams_only	LogReg, Lemma, Unigrams	75,90%	0,490
naive_bayes	MultinomialNB, Count-Vec	72,76%	0,508
random_forest	RandomForest, 5k Features	71,75%	0,384

Tabelle 10: Vergleich der Ergebnisse: 30.000 vs. 150.000 Datensätze

Experiment	Accuracy (30k)	Accuracy (150k)	Differenz
high_features	74,22%	76,77%	+2,55
baseline	74,29%	76,66%	+2,37
stemming_comparison	74,27%	76,61%	+2,34
svm_linear	73,60%	76,12%	+2,52
unigrams_only	73,98%	75,90%	+1,92
naive_bayes	72,18%	72,76%	+0,58
random_forest	70,04%	71,75%	+1,71

Tabelle 11: Confusion Matrix – Logistic Regression (Baseline, 150k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	2044	96	79	37	464
2 Sterne	442	193	124	60	325
3 Sterne	290	65	392	184	616
4 Sterne	92	15	122	680	1733
5 Sterne	157	15	62	273	13940

Tabelle 12: Confusion Matrix – Naive Bayes (150k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	1993	214	148	75	290
2 Sterne	429	290	178	75	172
3 Sterne	297	186	469	223	372
4 Sterne	121	93	265	982	1181
5 Sterne	351	164	249	1045	12638

Tabelle 13: Confusion Matrix – Random Forest (150k)

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	1214	2	0	2	1502
2 Sterne	194	96	1	4	849
3 Sterne	90	0	150	5	1302
4 Sterne	29	0	1	287	2325
5 Sterne	47	0	1	2	14397

Tabelle 14: Vergleich: Standard vs. Balanced Class Weights

Modell	**Acc. (normal)**	**Acc. (balanced)**	**F1 (normal)**	**F1 (balanced)**
Logistic Regression	76,66%	66,85%	0,507	0,504
SVM (LinearSVC)	76,12%	71,94%	0,502	0,504
Random Forest	71,75%	66,98%	0,384	0,464

Tabelle 15: Recall-Vergleich pro Klasse: Standard vs. Balanced (Logistic Regression)

Rating	**Recall (normal)**	**Recall (balanced)**	**Differenz**
1 Stern	75%	69%	-6
2 Sterne	17%	39%	+22
3 Sterne	25%	40%	+15
4 Sterne	26%	48%	+22
5 Sterne	96%	75%	-21

Tabelle 16: Confusion Matrix – Logistic Regression Balanced

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	1882	481	217	68	72
2 Sterne	326	446	249	72	51
3 Sterne	212	323	626	297	89
4 Sterne	103	168	452	1258	661
5 Sterne	413	416	603	2185	10830

Tabelle 17: Confusion Matrix – SVM Balanced

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	1915	349	219	59	178
2 Sterne	382	349	232	64	117
3 Sterne	256	268	548	249	226
4 Sterne	118	180	371	920	1053
5 Sterne	319	322	434	918	12454

Tabelle 18: Confusion Matrix – Random Forest Balanced

	Vorhergesagt				
Tatsächlich	1	2	3	4	5
1 Stern	2269	87	88	114	162
2 Sterne	548	177	127	144	148
3 Sterne	425	81	405	357	279
4 Sterne	296	55	173	1170	948
5 Sterne	1403	131	277	1587	11049