



BORIS THOME

NATURAL LANGUAGE PROCESSING WORKSHOP

ZIELSETZUNG

- ▶ Durchführen einer Klassifikation an einem Beispiel
- ▶ Wiederholung und Anwendung von Vorverarbeitungsmethoden
- ▶ Vorstellen einfacher Klassifikationsalgorithmen
- ▶ Auswertung der Ergebnisse

ÜBERSICHT

- ▶ Vorstellung des IMDB Datensatzes
- ▶ Vorverarbeitung der Textdaten
- ▶ Klassifikationsalgorithmen
- ▶ Codebeispiel
- ▶ Diskussion & Fragen

IMDb (INTERNET MOVIE DATABASE) – DATENSATZ

- ▶ Datensatz beinhaltet 50 000 Nutzerbewertungen zu Filmen
- ▶ Datensatz besteht aus Texten und zugehörigen Sentiment-Labels (Positiv / Negativ)
- ▶ Nutzerbewertung: Text und einer Punktbewertung
- ▶ Annotation erfolgte anhand der Punktbewertung der Nutzer:
 - ▶ Punktbewertung von 1 bis 10 möglich
 - ▶ **Negatives Sentiment:** Bewertung von 1 bis 4
 - ▶ **Positives Sentiment:** Bewertung von 7 bis 10
 - ▶ Maximal 30 Bewertungen pro Film

IMDb

(INTERNET MOVIE DATABASE) – BEISPIEL



Harry Potter und der Stein der Weisen (2001)
User Reviews
[+ Review this title](#)

1.979 Reviews

☐ Hide Spoilers

Filter by Rating:

Show All

Sort by:

Featured

↓↑

★ 10/10

There's nothing like the first
[HotToastyRag](#) 17 June 2019

There's nothing like the first in a series, is there? The introduction to the characters, the immersion into the fictional world, the first time you laugh, cry, care, and fear for someone's safety can never be repeated. No matter how many Harry Potter movies they crank out, or if they ever remake them in the future, none will come close to the wonderful first film, Harry Potter and the Sorcerer's Stone.

I'm sure everyone has their own childhood memories of reading the Harry Potter books that they'll tell their grandkids about, but I'll never forget going to see the first movie in the theaters. The lights dimmed, John Williams's perfect theme played its first notes as Richard Harris walked down Privet Drive, and everyone in the theater was transported to another world. John Williams's numerous themes, all wonderful and a personification of the wizarding world, took the early movies to another level. As other composers tried their hands at the later films, that quality was missing. There's something truly special about going to see this movie on the big screen, and while the "magical" qualities might not all be credited to the music, it's certainly one of them.

VORVERARBEITUNG VON TEXTDATEN

- ▶ Vorgehensweise hängt von mehreren Faktoren ab
 - ▶ Sprache:
 - ▶ Einsprachig / Mehrsprachig
 - ▶ Englisch / Deutsch /...
 - ▶ Textart:
 - ▶ Social Media Texte
 - ▶ Zeitungstexte
 - ▶ Aufgabe:
 - ▶ Sentiment-Analyse
 - ▶ Keyword Extraction

STOPPWÖRTER (STOP WORDS)

- ▶ Definition: **Stoppwörter** nennt man in der Informationsrückgewinnung bzw. im **Information Retrieval** Wörter, die bei einer **Volltextindexierung** nicht beachtet werden, da sie sehr häufig auftreten und gewöhnlich keine Relevanz für die Erfassung des **Dokumentinhalts** besitzen. (<https://de.wikipedia.org/wiki/Stopppwort>)
- ▶ Stoppwörter werden oft im Preprocessing herausgefiltert
- ▶ Grund: Sie tragen nicht zur semantischen Bedeutung von Texten bei, treten aber sehr häufig auf!
- ▶ Beispiele für Stoppwörter:
 - ▶ Bestimmte Artikel (the)
 - ▶ Unbestimmte Artikel (a, an)
 - ▶ Konjunktionen (and, or, ...)

ENTFERNEN VON STOPPWÖRTERN (STOP WORD REMOVAL)

- ▶ Implementierung mittels Wortlisten (z.B. Python NLTK)
- ▶ Iteration über alle Wörter im Text
- ▶ Vergleich mit vorgefertigter Wortliste
- ▶ Wird ein Stoppwort aus der Liste erkannt, so wird es aus dem Text entfernt
- ▶ Wortlisten können variieren
- ▶ Erweiterung der Listen um häufig auftretende Wörter möglich

ENTFERNEN VON STOPPWÖRTERN (STOP WORD REMOVAL)

► Englische Stoppwortliste von NLTK:

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

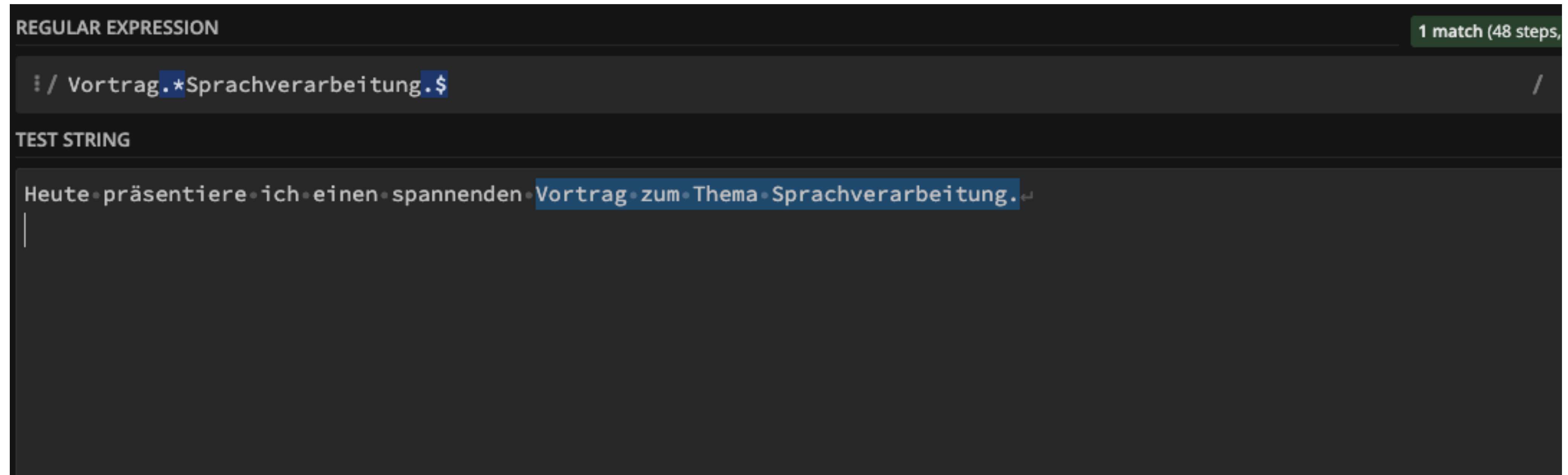
LEMMATISIERUNG (LEMMATIZATION)

- ▶ Reduktion der flektierten Wortform auf die Grundform des Wortes
- ▶ Führt zur Vereinheitlichung der Daten:
 - ▶ Beispiel: „plays“, „played“, „playing“ => „play“
- ▶ Lemmatisierung meist mithilfe von Wörterbüchern (Lookup-Based)
- ▶ Problem: Wörterbuch/Liste enthält in der Regel nicht alle Wörter
- ▶ Kann beispielsweise mit spaCy in Python implementiert werden

REGULAR EXPRESSIONS (REGEX)

- ▶ Beschreibung von Mengen von Zeichenketten mithilfe syntaktischer Regeln
- ▶ Konzept stammt aus der theoretischen Informatik
- ▶ Suche komplexer Patterns (Sequenzen)
- ▶ Suchen und Ersetzen Funktion
- ▶ Spezifische Syntax notwendig
- ▶ In viele Programmiersprachen integrierbar
- ▶ Hilfsseiten zum Testen von Regex Befehlen (<https://regex101.com/>)

REGULAR EXPRESSIONS (BEISPIEL)



The screenshot shows a web-based regular expression testing interface. At the top, the label 'REGULAR EXPRESSION' is on the left, and '1 match (48 steps)' is on the right. Below this, the regular expression pattern `/Vortrag.*Sprachverarbeitung.$` is entered in a text field. Underneath, the label 'TEST STRING' is on the left, and the test string `Heute präsentiere ich einen spannenden Vortrag zum Thema Sprachverarbeitung.` is entered in a text area. The words 'Vortrag' and 'Sprachverarbeitung.' in the test string are highlighted with a blue background, indicating a successful match.

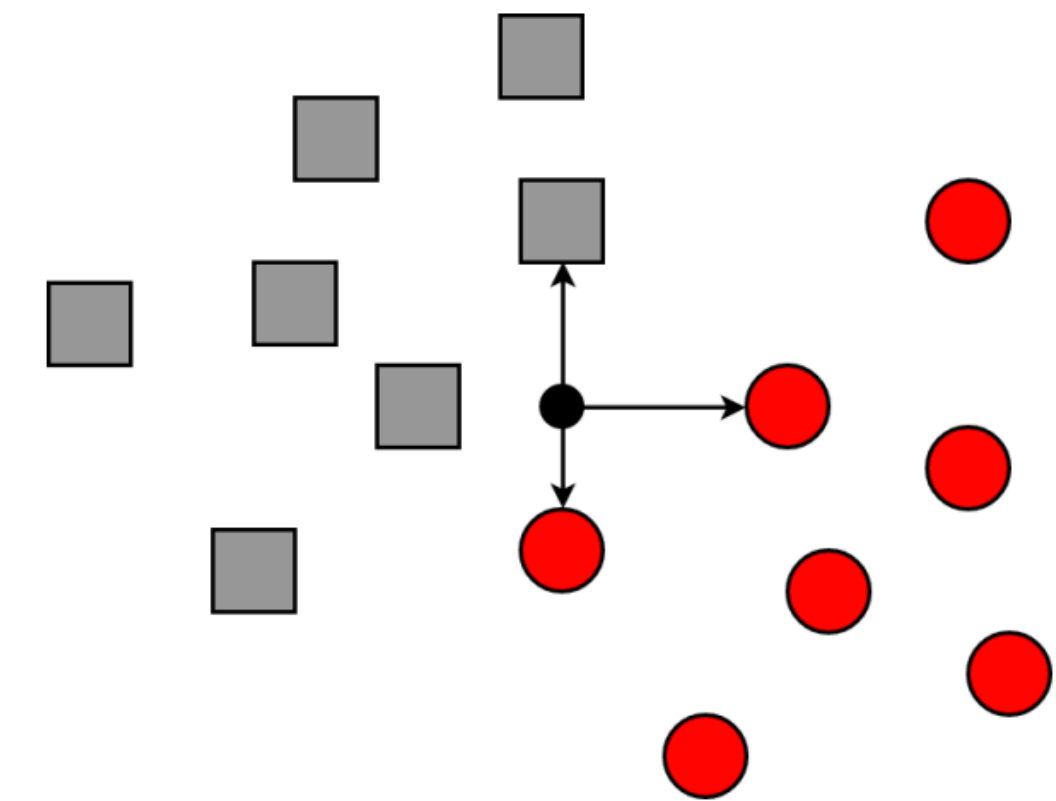
- ▶ Zeichenkette, die mit „Vortrag“ beginnt und mit „Sprachverarbeitung.“ Endet
- ▶ Zwischen den beiden Worten dürfen beliebige Zeichen stehen „.*“

VORKOMMENSHÄUFIGKEIT – INVERSE DOKUMENTHÄUFIGKEIT (TF-IDF)

- ▶ Kombination aus Vorkommenshäufigkeit und Inverser Dokumenthäufigkeit
- ▶ $\text{idf}(t) = \log \frac{N}{\sum_{D:t \in D} 1}$ (Inverse Dokumenthäufigkeit)
 - ▶ (N : Anzahl aller Dokumente im Korpus, t : Term, D : Dokument)
- ▶ $\text{tf.idf}(t, D) = \text{tf}(t, D) \cdot \text{idf}(t)$ (Term frequency - Inverse Document Frequency)
 - ▶ Multiplikative Verknüpfung mit Vorkommenshäufigkeit
- ▶ Übertragen in ein geeignetes Format für das Training eines ML Algorithmus

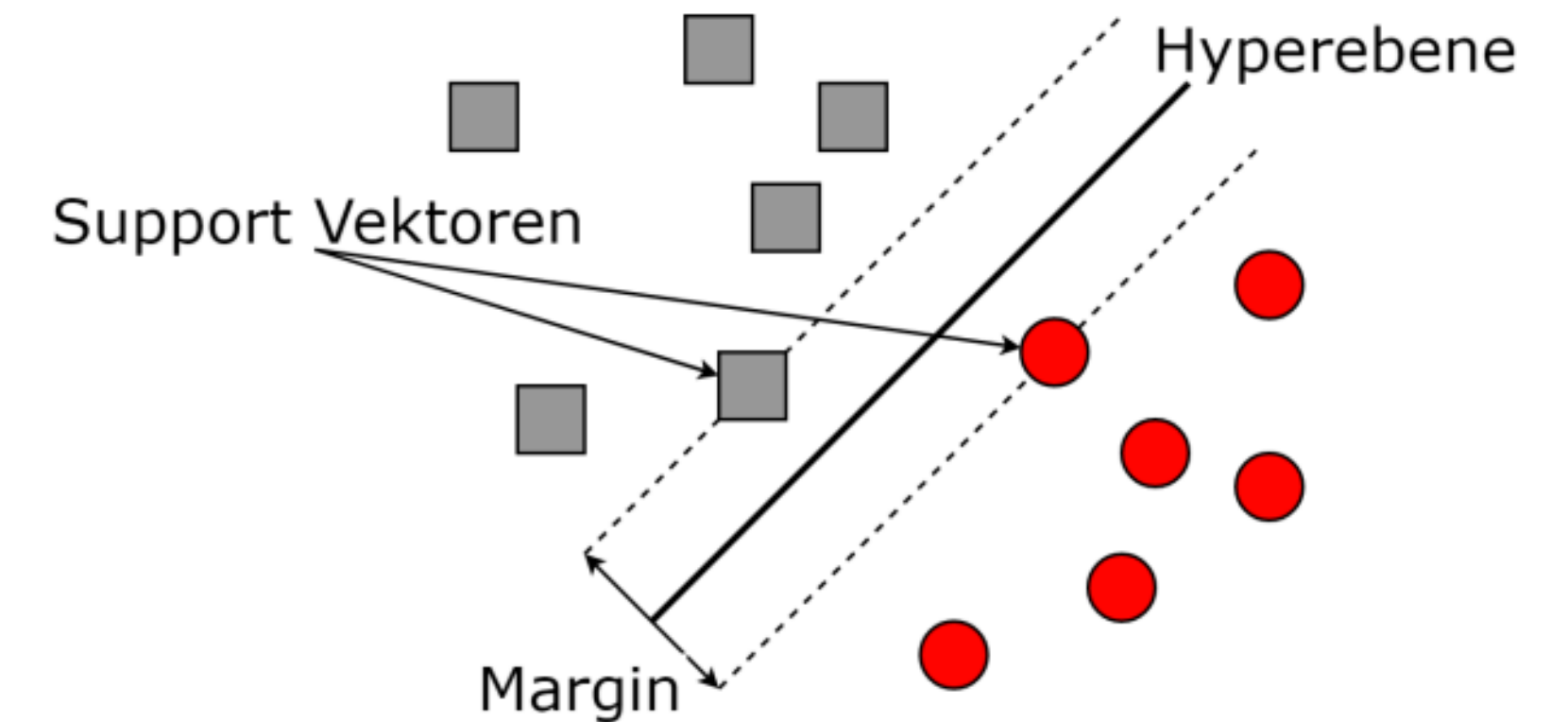
K-NEAREST-NEIGHBORS (KNN)

- ▶ Speichert Datenpunkte des Trainingsdatensatzes
- ▶ Betrachtung der k nächsten Datenpunkte
- ▶ Klassenzuteilung erfolgt anhand eines Mehrheitsentscheids
- ▶ Abbildung zeigt eine binäre Klassifikation mit $k=3$
- ▶ Bei binärer Klassifikation empfiehlt sich ein ungerades k für den Mehrheitsentscheid



SUPPORT VECTOR MACHINES (SVM)

- ▶ Input: Menge von Objekten und deren Klassenzugehörigkeit
- ▶ Die beiden nächsten Datenpunkte der unterschiedlichen Klassen werden als „Support Vektoren“ bezeichnet
- ▶ Ziel des Trainings:
 - ▶ Hyperebene mit möglichst großen Abstand (Margin) zu den Support Vektoren finden



K-FOLD-CROSS-VALIDATION

- ▶ Ziel: Testdatenmenge maximieren ohne Überanpassung im Trainingsprozess
- ▶ Aussagekräftigere Auswertung durch Aufteilung von Trainings- und Testdaten
- ▶ Aufteilung der Daten in k folds (Falten)
- ▶ Beispiel $k=5$
 - ▶ Training auf 80% und Testing auf 20%
 - ▶ Wiederhole diesen Prozess 5 mal
 - ▶ Testing auf 5 disjunkten Datenmengen
 - ▶ Bestimmung der Mittelwerte



BEISPIELCODE



Q & A