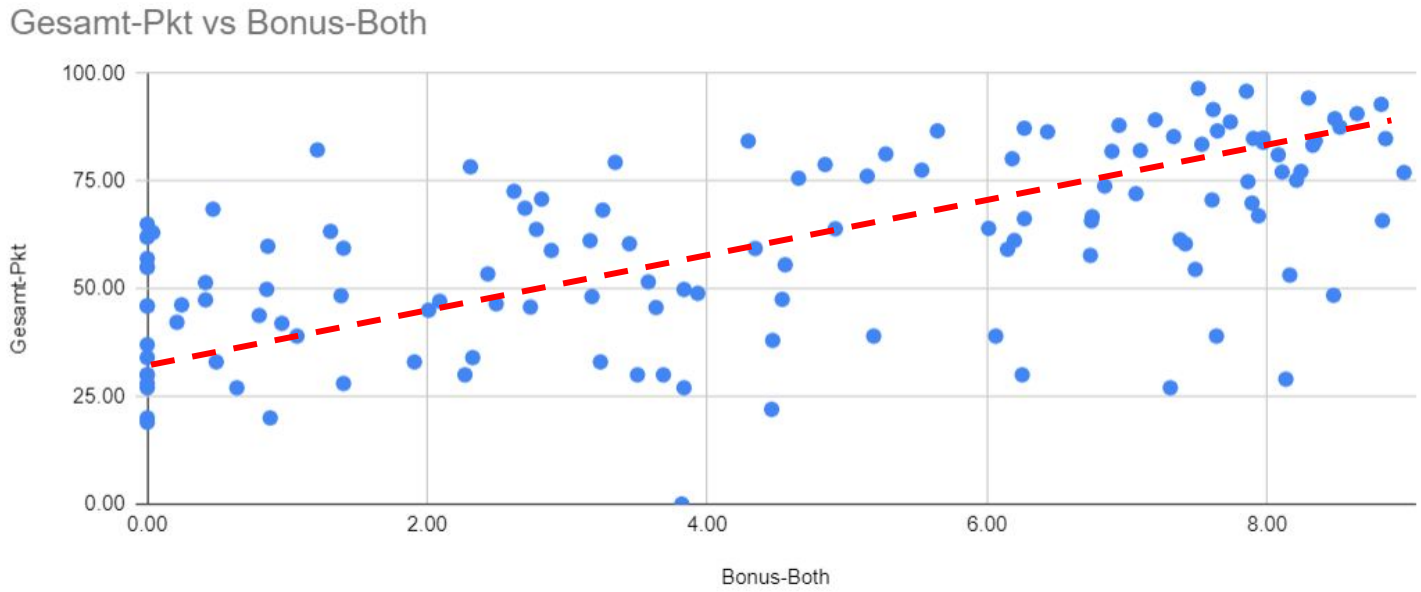
Computer-Linguistische Anwendungen

CLA | B.Sc. | LMU

Hinweise: Übungen /



Vorlesung:
Paraphrasen-Erkennung, NumPy, Scikit-Learn

Philipp Wicke, PhD
Centrum für Sprach- und Informationsverarbeitung
Ludwig-Maximilians-Universität München
pwicke@cis.lmu.de

30”

Übersicht

- Paraphrasen-Erkennung
- Daten-Repräsentationen für maschinelles Lernen
- Einführung in Numpy
- Scipy Sparse Matrices
- Scikit-Learn Vectorizer
- SOTA: Sentence Transformer

Paraphrasen-Erkennung

Paraphrasen-Erkennung

- Ist ein Satz (A) eine Paraphrase eines anderen Satzes (B)?
- Enthalten zwei Tweets die gleiche Information?
- Dies ist **nicht** einfach zu entscheiden
  - Was ist eine Paraphrase?
  - Gibt es überhaupt Paraphrasen, deren Bedeutung zu 100% gleich ist?
    Paraphrase , starke Ähnlichkeit, annähernd gleiche Bedeutung
  - Sprachliche Variabilität: derselbe Sachverhalt kann ganz verschieden ausgedrückt werden
  - Besonders schwierig bei Twitter-Daten: Abkürzungen, Rechtschreibfehler ...
- Beispiele:

(A)	I hate the Orange POTU$, he drives me madddd
(B)	Idc., but donald trump does not align with my values, I do not like him.

(A)	There is a storm warning in Chicago. Bring umbrellas!!
(B)	It's always sunny in Philadelphia is my favorite TV show, period.

SemEval-2015 Task 1: Paraphrase Similarity on Twitter

SemEval is a series of international natural language processing (NLP) research workshops whose mission is to advance the current state of the art in semantic analysis and to help create high-quality annotated datasets in a range of increasingly challenging problems in natural language semantics (https://semeval.github.io/)

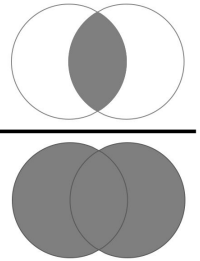
Task 1:

- ca. 19,000 Tweet-Paare, die über Amazon Mechanical Turk annotiert wurden
- Binäre Klassifikation: Eine Tweet-Paar ist eine Paraphrase (True) oder nicht (False)
- **Brainstorming**: was sind gute Merkmale zur Paraphrasenerkennung?

Merkmale zur Paraphrasen-Erkennung I

- Wort-Überlappung (Word overlap)
  - Einfachste Variante: Anzahl der gemeinsamen Wörter (bzw. Token) in beiden Tweets (ohne deren Häufigkeit zu berücksichtigen).

    “overlap”



  - Benötigte Normalisierung (längere Tweets haben mehr Wörter, sind aber nicht unbedingt ähnlicher). Einfache Lösung: Zusätzliche Merkmale für die Anzahl der verschiedenen Wort-Types in Text1 und Text2.

    “union”
- Wort-Ngram-Überlappung (Word-Ngram overlap)
  - Reihenfolge der Wörter wird berücksichtigt.
  - Ansonsten wie für Wort-Überlappung.
  - 3-Gramme funktionieren gut für diese Aufgabe.

Merkmale zur Paraphrasen-Erkennung II

- Wort-Paar Merkmale (Word-pair features)

    - Wie können wir Paraphrasen erkennen, die unterschiedliche, aber semantisch ähnliche Wörter verwenden?
    - Das Modell soll aus den Trainingsdaten für möglichst viele Wortpaare lernen, ob die Wörter ähnlich sind!
    - Jede Kombination aus einem Wort aus `Text1` mit einem Wort aus `Text2` ist ein Merkmal

Beispiel: Merkmals-Repräsentation

(A) happy towel day, have a happy 25th of May!
(B) Don’t forget your towels, have a happy towel day!

```
{    "word_overlap": ,
    "three_gram_overlap": ,
    "word_union": ,
    "threegram_union": ,

     …                          }
```

Beispiel: Merkmals-Repräsentation

(A) happy towel day, have a happy 25th of May!
(B) Don't forget your towels, have a happy towel day!

```
{    "word_overlap": 5,
    "three_gram_overlap": 2,
    "word_union": ,
    "threegram_union": ,

     …                          }
```

Beispiel: Merkmals-Repräsentation

(A) happy towel day, have a happy 25th of May
(B) Don't forget your towels, have a happy towel day

```
{     "word_overlap": 5,
      "three_gram_overlap": 2,
      "word_union": 12,
      "threegram_union": ,

      …                                }
```

Beispiel: Merkmals-Repräsentation

(A) happy towel day, have a happy 25th of May
(B) Don't forget your towels, have a happy towel day

7
+ (7 - 2)
=
12


```
{    "word_overlap": 5,
    "three_gram_overlap": 2,
    "word_union": 12,
    "threegram_union": 12,

     …                          }
```

Beispiel: Merkmals-Repräsentation

(A) happy towel day, have a happy 25th of May
(B) Don’t forget your towels, have a happy towel day

7
+ (7 - 2)
=
12


```
{    "word_overlap": 5,
    "three_gram_overlap": 2,
    "word_union": 12,
    "threegram_union": 12,
    "happy#don’t": 1,
    "happy#forget": 1,
    "towel#don’t": 1,
     …                          }
```

Implementierung

Was ist das Ergebnis dieser List-Comprehension?

```
l=["wishing", "everyone", "a", "happy", "memorial", "day"]
n=2
[l[i:i+n] for i in range(len(l)-n+1)]
```

Implementierung

Was ist das Ergebnis dieser List-Comprehension?

```
l=["wishing", "everyone", "a", "happy", "memorial", "day"]
n=2
[l[i:i+n] for i in range(len(l)-n+1)]
```

```
[['wishing', 'everyone'],
 ['everyone', 'a'],
 ['a', 'happy'],
 ['happy', 'memorial'],
 ['memorial', 'day']]
```

Wie können Wort-Paar Merkmale implementiert werden?

Daten-Repräsentationen

für maschinelles Lernen

Daten

- Datensatz: Sammlung von Instanzen
- Design-Matrix

$$\mathbf{X} \in \mathbf{R}^{n \times m}$$

  - ➢ n: Anzahl von Instanzen
  - ➢ m: Anzahl von Merkmalen
  - ➢ Beispiel: $X_{i,j}$ Wert des Merkmals j (z.B. Häufigkeit eines Wortes) in Dokument *i*

Bei überwachten (supervised) Lernverfahren, z.B. Klassifizierung, gibt es für jede Instanz noch ein **Label**

  - ➢ Label: vorauszusagender Wert/Kategorie
  - ➢ Trainingsdaten: Labelvektor $\mathbf{y} \in \mathbf{R}^n$ zusätzlich zu **X**

Jede Zeile in Matrix $\vec{X}$ und Vektor $\vec{y}$ entspricht einer Instanz

$$\vec{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix} \quad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Binäre Klassifikation:

$$\vec{y} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \text{ ... oder ... } \begin{bmatrix} -1 \\ 1 \\ \vdots \\ -1 \end{bmatrix}$$

Multi-Klassen Klassifikation (*one-hot-encoding*):

$$\vec{Y} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ \vdots & & \\ 0 & 1 & 0 \end{bmatrix}$$

Daten-Repräsentationen

- Für eine bessere Performanz (Geschwindigkeit, Speicherbedarf) verwenden Machine-Learning Toolkits (scikit-learn, Keras, ...) Matrizen (statt z.B. string-to-count Dictionaries).

- Spezielle Module und Klassen zur Matrizen-Rechnung stellen effiziente Implementationen bereit, z.B. für
  - mathematische Operatoren (Matrizen-Multiplikation, Vektor-Addition...)
  - Datenzugriff und -umwandlung (Zugriff auf Zeilen/Spalten, Transponieren einer Matrix)

- Was wäre eine angemessene Datenstruktur für die folgenden Merkmalsräume:
  - Jedes Merkmal ist ein Grauwert in einem 100 × 100 Pixel großen Schwarz-Weiß Bild?
  - Jedes Merkmal ist ein Indikator, ob ein bestimmtes Wort in einem Dokument vorkommt (Vokabular Größe 10,000)?

Daten-Repräsentationen

- Was wäre eine angemessene Datenstruktur für die folgenden Merkmalsräume:
  - Jedes Merkmal ist ein Grauwert in einem 100 × 100 Pixel großen Schwarz-Weiß Bild?

    Fast alle Pixel haben einen unterschiedlichen Wert 6= 0. Alle Werte müssen abgespeichert werden z.B. in einer Liste, oder einem Array (Werte werden in einem kontinuierlichen Speicherbereich abgelegt).

    → **Numpy Arrays**

  - Jedes Merkmal ist ein Indikator, ob ein bestimmtes Wort in einem Dokument vorkommt (Vokabular Größe 10,000)?

    Die meisten Merkmale haben den Wert 0. Eine angemessene Datenstruktur speichert nur die Werte, die ungleich 0 sind. (Z.B. mit einem Dictionary: (Zeile, Spalte) → Wert.)

    → **SciPy Sparse Matrices**