Introduction to Text Mining and Natural Language Processing

Session 1: Introduction to Text Mining, Strings and Regex

Hannes Mueller

January 2026

Barcelona School of Economics

Introduction

Introduction

- Logistics of this Course
- Language and Cognition
- (Text Mining) Project Management
- Definitions and Method Overview
- Strings and Regularized Expressions

Logistics of this Course

NLP in the age of GenAI

What should we focus on in an age of GenAI?

Answer here:

1) project management

2) technical skills (bugfixing the vibe-coders)

3) integrating project management and technical knowledge

Course Overview

Part 1: Project Design and Getting the Text (Session 2)

Part 2: Text Mining Basics (Sessions 3 and 4)

Part 3: Dimensionality Reduction (Sessions 5 to 7)

In-class assignment in session 6

Part 4: Supervised Learning with Text (Sessions 8 and 9)

In-class assignment in session 9

Write me your term-paper ideas throughout.

Term paper presentations: 16th of March (feedback!)

Grading

Simple methods but **integration is important**. TAs will reinforce this.

In-class assignments 30%, Term paper presentation 20%, Term paper 50%.

Instructions are on class page. Key elements:

- exercises from session to session
- exercises feed 2 in-class assignments (15 minutes only)
- 1 term-paper plan in self-selected groups
- 1 term-paper (min. 5 pages)

Language and Cognition

Language and Cognition

Text is a written representation of language. The importance of language is hard to overstate:

- Wittgenstein argued that the world we see is defined and given meaning by the words we choose.
- The Sapir–Whorf hypothesis in linguistics (contested!) states that the grammatical structure of a mother language influences the way we perceive the world.
- Boroditsky (2011) discusses some striking examples (spatial orientation, perception of time, memory).
- Chen (2013) shows that the languages that grammatically associate the future and the present, foster future-oriented behavior.

Language and Cognition

The label-feedback hypothesis of Lupyan (2012): language as a modulator of a distributed and interactive system. Language shapes cognition dynamically, not by irrevocably re-wiring how we see the world.

Fedorenko et al (2024): Language is primarily a tool for communication rather than thought. Language co-evolved with our thinking and reasoning capacities, and only reflects, rather than gives rise to, the signature sophistication of human cognition.

LLMs and Brains

- Autoregressive LMs and the brain share a strong emphasis on context-dependent prediction.
- Contextual embeddings from modern LMs can predict neural responses during naturalistic language comprehension.
- As LMs improve on next-word prediction, their representations often become **more brain-aligned**.

Take-away for Text Mining

- Text is a behavioral trace: it reflects language computations plus goals, incentives, audience design, and culture.
- Language can modulate perception and memory *in the moment*, so word choices can shape what becomes salient.
- Modern NLP reconstructs latent structure from usage patterns (topics, frames, stance, style) — but this structure is not identical to "thought".
- **Your most important tool remains human judgment:** defining constructs, validating labels, and interpreting results in context.
    - You can see this in how people typically show performance by showing language output.

(Text Mining) Project Management

The Goal of the Course

- Prepare you to integrate text into project work.

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
    - get texts from somewhere

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
  - get texts from somewhere
  - compile text into a useful data format

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
  - get texts from somewhere
  - compile text into a useful data format
  - clean text and pre-process text

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
  - get texts from somewhere
  - compile text into a useful data format
  - clean text and pre-process text
  - explore text descriptively

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
  - get texts from somewhere
  - compile text into a useful data format
  - clean text and pre-process text
  - explore text descriptively
  - use text for prediction

The Goal of the Course

- Prepare you to integrate text into project work.
- When you work for DM support you might be asked to do some of the following:
  - get texts from somewhere
  - compile text into a useful data format
  - clean text and pre-process text
  - explore text descriptively
  - use text for prediction
  - use text for statistical inference

Data Science's Biggest Enemy

- Machine learning is fun and powerful.

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.
- Projects often face three problems:

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.
- Projects often face three problems:
    - Not enough time spent on defining the problem: leads to confusion.

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.
- Projects often face three problems:
  - Not enough time spent on defining the problem: leads to confusion.
  - Large datasets tempt into showing impressive viz that have little real value.

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.
- Projects often face three problems:
  - Not enough time spent on defining the problem: leads to confusion.
  - Large datasets tempt into showing impressive viz that have little real value.
  - Small data: the smallest common denominator is what determines your data size in projects using supervised methods.

Data Science's Biggest Enemy

- Machine learning is fun and powerful.
- Cross validation is a safeguard against overfitting in supervised problems.
- Projects often face three problems:
  - Not enough time spent on defining the problem: leads to confusion.
  - Large datasets tempt into showing impressive viz that have little real value.
  - Small data: the smallest common denominator is what determines your data size in projects using supervised methods.
- You need to use your human prior. Make it as explicit as possible.

Integration

- Everything needs to start with a goal or research question.

Integration

- Everything needs to start with a goal or research question.
- Read literature and build conceptual model in your head/on paper.

Integration

- Everything needs to start with a goal or research question.
- Read literature and build conceptual model in your head/on paper.
- Data projects are a back-and-forth between data work and conceptual thinking.

Integration

- Everything needs to start with a goal or research question.
- Read literature and build conceptual model in your head/on paper.
- Data projects are a back-and-forth between data work and conceptual thinking.
    - Theory without empirical test is meaningless, empirical test without theory is impossible.

Integration

- Everything needs to start with a goal or research question.
- Read literature and build conceptual model in your head/on paper.
- Data projects are a back-and-forth between data work and conceptual thinking.
    - Theory without empirical test is meaningless, empirical test without theory is impossible.
- If you have a theory your data will talk to you.

Integration

- Everything needs to start with a goal or research question.
- Read literature and build conceptual model in your head/on paper.
- Data projects are a back-and-forth between data work and conceptual thinking.
    - Theory without empirical test is meaningless, empirical test without theory is impossible.
- If you have a theory your data will talk to you.
- This then allows you to really learn something beyond the data.

Examples from Research

Predict armed conflict using news articles

- outbreaks are rare, hard to predict, small data
- topic model used for summarizing articles
- how many topics should we choose?

Event detection in a large news corpus

- rare events (less than 0.1% of articles)
- 300 hours of hand-coding
- subtle language understanding required
- how to proceed?

Introduction to Strings and RegEx

What Are Strings?

- Strings are sequences of characters used to store text data.
- Enclosed in quotes: single (' '), double (" "), or triple ("' "' or """ """).
- Used for storing and manipulating textual information.

Characteristics of Python Strings

- **Immutability:** Once created, strings cannot be changed.
- **Indexing:** Access individual characters using indexes.
  - First character: `string[0]`
  - Last character: `string[-1]`

Useful String Operations

- **Concatenation:** Combining strings.
  - Example: `'Hello' + ' ' + 'World'`
- **Length:** Finding the number of characters.
  - Example: `len('Hello')` outputs 5
- **Slicing:** Extracting parts of the string.
  - Example: `'Hello'[1:4]` outputs 'ell'

More Useful String Operations

- **Query for content:** Operators work as in lists
  - Example: `"w" in string` gives `True` if fulfilled
- **Lowercasing:** Set all letters lower case.
  - Example: `string="WHAT you want"` and `string.lower()` outputs `what you want`
- **Find position:** Find the first position of an occurrence
  - Example: `string.find("want")` outputs 9

Key method: split()

- The `split()` method in Python is used to split a string into a list.
- By default, it splits the string by whitespace but can split using any other delimiters when specified.
- Syntax: `string.split(separator, maxsplit)`

Example:

```
text = "Hello, welcome to the world of Python"

words = text.split()

print(words)
```

This will output:

['Hello,', 'welcome', 'to', 'the', 'world', 'of', 'Python']

What is RegEx?

- Regular Expressions (RegEx) are sequences of characters that form a search pattern.
- Used for pattern matching, searching, and string manipulation.
- Fundamental in text processing and data extraction.
- There are a lot of special characters and functions. Best seen "in action".
- We will briefly introduce concepts and then move on to a Jupyter notebook.

Basic Syntax of RegEx

- **Literals:** Match exact characters. For example, `abc` matches "abc".
- **Metacharacters:** Symbols with special meanings. For example, `.` matches any character, `^` matches the start of a string.
- **Quantifiers:** Specify the number of occurrences. For example, `*` for 0 or more, `+` for 1 or more.

RegEx Functions

- `re.findall(pattern, string)`
  - Returns all matches of the pattern in a string as a list.
  - **Example:** `re.findall(r'\d+', 'A1 B22 C333')`
    Output: `['1', '22', '333']`
- `re.sub(pattern, replacement, string)`
  - Replaces all matches of the pattern with a replacement string.
  - **Example:** `re.sub(r'\d+', 'X', 'ID: 123')`
    Output: "ID: X"

Groups and Capturing in RegEx

- Use parentheses to create groups in patterns.
- Groups can extract parts of the string for further processing.
- **Example 1:**
  - Pattern: `(\d{3})-(\d{2})-(\d{4})`
  - Matches: "123-45-6789" (e.g., a U.S. Social Security Number)
  - Captures three groups: "123", "45", "6789"
- **Example 2:**
  - Pattern: `\d{2}/\d{2}/\d{4}`
  - Matches: "12/02/2023"
  - Captures the three elements of the date.

Lookahead and Lookbehind in RegEx

- Lookahead and lookbehind assert what should/shouldn't follow or precede a match.
- They don't consume characters in the string, but only assert whether a match is possible or not.
- **Positive Lookahead Example:**
  - Pattern: `q(?=u)`
  - Matches: "q" in "quick" (only if it's followed by "u")
- **Negative Lookbehind Example:**
  - Pattern: `(?<!q)u`
  - Matches: "u" in "guru" but not in "quick"

Non-Greedy Matching in RegEx

- Non-greedy (or lazy) matching finds the smallest possible match.
- This is useful when you want to match as little as possible.
- **Example:**
  - Greedy Pattern: `<.*>`
    - Matches: The entire string "`<div>Hello <span>World`"
  - Non-Greedy Pattern: `<.*?>`
    - Matches: "`<div>`" and "`<span>`" separately in "`<div>Hello<span>World`"