Introduction

PU Tools, Ressourcen, Infrastruktur

Nils Reiter,
nils.reiter@uni-koeln.de

November 5, 2020
(Winter term 2020/21)

Nils Reiter

- Vertretungsprofessor Sprachliche Informationsverarbeitung / Digital Humanities (seit September 2019)
- Davor: Uni Stuttgart, Uni Heidelberg, Uni Saarbrücken
- Studium Computerlinguistik / Informatik
- Forschungsinteressen
  - Angewandte Sprachtechnologie
  - Computational Literary Studies
  - Formalisierung / Operationalisierung komplexer Probleme

Ideen für ein erträgliches Online-Semester (lessons learned)

- Synchronveranstaltung
  - Interaktion zwischen und Ihnen untereinander sowie Ihnen und mir
- Gruppenarbeit(en)
  - Verteilen Sie sich in den Räumen auf dem Discord-Server
- kein Ilias-Forum mehr ☺

Zum Warmwerden: Vorstellungsrunde

Drei Fragen

- Wer seid Ihr?
- Welche Technik würdet Ihr gerne lernen? / Was interessiert Euch besonders?
- Was macht Ihr als erstes, “wenn Corona vorbei ist”?

Kursablauf

In jeder Sitzung:

- Einführung in neues Thema
- Vorstellung der Übungsaufgabe
- Lab session: Arbeit in Kleingruppen an der Übung
- Fertigstellen der Übung bis Mittwochabend der nächsten Woche
  - Abgabe der Übungen in einem eigenen branch auf via GitHub
- Kommentierte Referenzlösung erscheint als update via GitHub

Kursablauf

Lernziele

- Code mit git versionieren
- Ein Java-Projekt mit maven verwalten
- Einfache *machine learning*-Experimente mit Weka und mallet durchführen
- NLP-Pipelines in Apache UIMA erstellen

Kursablauf

Lernziele

- Code mit git versionieren
- Ein Java-Projekt mit maven verwalten
- Einfache *machine learning*-Experimente mit Weka und mallet durchführen
- NLP-Pipelines in Apache UIMA erstellen

Studienleistung

- Jede Woche eine Übung committen
  - Stand ist in ilias einsehbar

Kursorganisation

Ressourcen, Literatur, Kommunikation

- Kurswebseite `https://lehre.idh.uni-koeln.de/lehrveranstaltungen/wisem20/pu-tools-ressourcen-infrastruktur/`
  - Folien, Zeitplan
- GitHub-Gruppe: `https://github.com/idh-cologne-tools-ressourcen-infra`
  - Übungen erscheinen dort als Repository
  - Manchmal gibt es ein Update für ein existierendes Repository

Kursorganisation

Ressourcen, Literatur, Kommunikation

- Kurswebseite `https://lehre.idh.uni-koeln.de/lehrveranstaltungen/wisem20/pu-tools-ressourcen-infrastruktur/`
  - Folien, Zeitplan
- GitHub-Gruppe: `https://github.com/idh-cologne-tools-ressourcen-infra`
  - Übungen erscheinen dort als Repository
  - Manchmal gibt es ein Update für ein existierendes Repository
- Discord
  - Sitzungen
  - Server kann auch außerhalb der Sitzungen benutzt werden
- E-Mail `nils.reiter@uni-koeln.de`
  - Alles andere

Modulprüfung

Ilias: BA-AM1-Angewandte-Linguistische-Datenverarbeitung.pdf

- Thema
  - Findung und Wahl: Ihre Aufgabe
  - Kann, muss aber nicht, etwas mit dem Seminar zu tun haben
  - Mit mir absprechen
- Praktischer Anteil: Offen.
  Beispiele: Experiment zur automatischen Identifikation eines Textphänomens, Annotationsexperiment, quantitativer Vergleich verschiedener Korpora, ...
- Am Ende: Hausarbeit von ca. 10 S. Länge
- 'Letzte' Übung vor der Bachelor-Arbeit
- Experiment: Reden über Ideen für Modulprüfungsthemen am 03.12. (im Hauptseminar)

Section 3

Overview

Version control

- Versioning of source code
- Differences between versions
- Maintaining several branches in parallel

Version control

- Versioning of source code
- Differences between versions
- Maintaining several branches in parallel

Why is this useful?

- Programming projects quickly become massive
  - Windows 2000: 28mio LoC (ca. 930k standard pages)
  - CorefAnnotator: 27k LoC (ca. 770 standard pages)

Version control

- Versioning of source code
- Differences between versions
- Maintaining several branches in parallel

Why is this useful?

- Programming projects quickly become massive
  - Windows 2000: 28mio LoC (ca. 930k standard pages)
  - CorefAnnotator: 27k LoC (ca. 770 standard pages)
- Large teams
  - working on the same project
  - over a long time (don't rely on human memory)

Version control

- Versioning of source code
- Differences between versions
- Maintaining several branches in parallel

Why is this useful?

- Programming projects quickly become massive
  - Windows 2000: 28mio LoC (ca. 930k standard pages)
  - CorefAnnotator: 27k LoC (ca. 770 standard pages)
- Large teams
  - working on the same project
  - over a long time (don't rely on human memory)
- A single conceptual change often distributed over many files

Maven

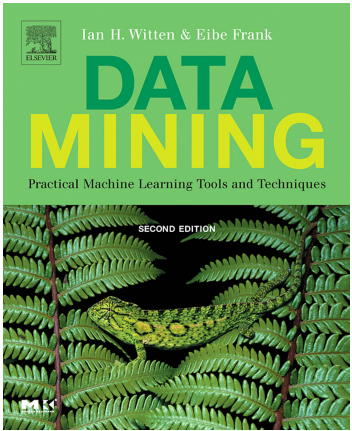
- Build manager: compilation, resource creation, packaging
- Dependencies
  - Defining external libraries
  - Downloading them from repositories
  - Including them on build path
- Example: `https://github.com/nilsreiter/CorefAnnotator/blob/master/pom.xml`

Maven

- Build manager: compilation, resource creation, packaging
- Dependencies
  - Defining external libraries
  - Downloading them from repositories
  - Including them on build path
- Example: `https://github.com/nilsreiter/CorefAnnotator/blob/master/pom.xml`

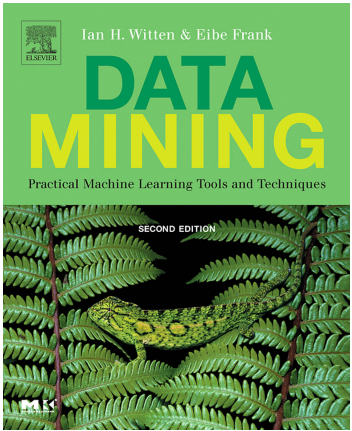
```
<project xmlns="http://maven.apache.org/POM/4.0.0">
 <groupId>de.unistuttgart.ims</groupId>
 <artifactId>coref.annotator</artifactId>
 <version>2.0.0-beta3</version>
 <name>CorefAnnotator</name>
 <dependencies>
  <dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-csv</artifactId>
  <version>1.8</version>
 </dependency>
</project>
```

Machine Learning with Weka



- Ian H. Witten and Eibe Frank (2005). *Data Mining*. 2nd ed. Practical Machine Learning Tools and Techniques. Elsevier
- `https://www.cs.waikato.ac.nz/ml/weka/`

Machine Learning with Weka



- Ian H. Witten and Eibe Frank (2005). *Data Mining*. 2nd ed. Practical Machine Learning Tools and Techniques. Elsevier
- `https://www.cs.waikato.ac.nz/ml/weka/`
- Collection of machine learning algorithms
- Preprocessing
- Graphical user interface: Good for exploration, bad for large data analysis
- Command line user interface, Java API: Bad for exploration, good for large data analysis
- Visualization options

Demo: Weka in action

`http://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set`

Mallet and LDA

- Latent Dirichlet allocation (LDA): Unsupervised method to detect topical structures in texts
- Mallet: Machine learning toolkit for LDA (and others)
  - http://mallet.cs.umass.edu/index.php
- Java API and command line interface

Deep learning

- New trend in machine learning: Artificial neural networks
  - New level of performance for NLP tools
- Deeplearning4J: Java API
  - https://deeplearning4j.org
- Session
  - A bit more theory, exercise over the break

Linguistic Data Structures

- What do we need to encode linguistic structures?
  - PoS tags, dependency parses, coreference chains, semantic roles, ...
- What do we need for applications on textual data?
  - Typed annotations and feature structures

Apache UIMA

- 'Unstructured Information Management Architecture'
  - `https://uima.apache.org`
  - Open source, Apache license
  - Originally developed at IBM
- Framework for
  - storing annotations (with efficient retrieval)
  - processing text and creating/consuming annotations
  - re-using components with clearly defined interfaces

Apache UIMA

- 'Unstructured Information Management Architecture'
  - `https://uima.apache.org`
  - Open source, Apache license
  - Originally developed at IBM
- Framework for
  - storing annotations (with efficient retrieval)
  - processing text and creating/consuming annotations
  - re-using components with clearly defined interfaces

dkpro

- Data structure definitions for common NLP phenomena
- Collection of UIMA components for various NLP tasks
  - Exactly: wrappers around other components
- `https://dkpro.github.io`

Other Topics

(if time permits)

- Tabular data
- Unit-testing
- ...?

Exercise 1

`https://github.com/idh-cologne-tools-ressourcen-infra/exercise-01`

References I

Witten, Ian H. and Eibe Frank (2005). *Data Mining*. 2nd ed. Practical Machine Learning Tools and Techniques. Elsevier.