DATA.STAT.840

STATISTICAL METHODS FOR TEXT DATA ANALYSIS

Jaakko Peltonen, Fall 2022

Course description

- This course teaches various statistical methods for modeling and analysing text data.
- Contents include
  - models for representing text including vector space models and neural embedding models;
  - document content processing stages such as lemmatization and keyphrase extraction;
  - probabilistic models of content variation including n-grams and topic models;
  - methods for various text analysis tasks.
- Learning outcomes: The student is familiar with document representation, modeling, and analysis techniques and can apply several basic techniques in text analysis tasks.

Topics

Contents are in development and may be adjusted as the course progresses. Some topics may be divided over multiple lectures.

1. Preliminaries
2. Document preprocessing, collocation/keyphrase extraction
3. Vector space models, document clustering and classification
4. Language models 1: N-gram models
5. Language models 2: Topic models
6. Language models 3: Hidden markov models
7. Sentiment analysis
8. Language models 3: Probabilistic context free grammars
9. Question answering
10. Word embedding and document visualization
11. Language models 4: Neural language models
12. Language model based information retrieval
13. Language model based Interactive information retrieval

Practical information

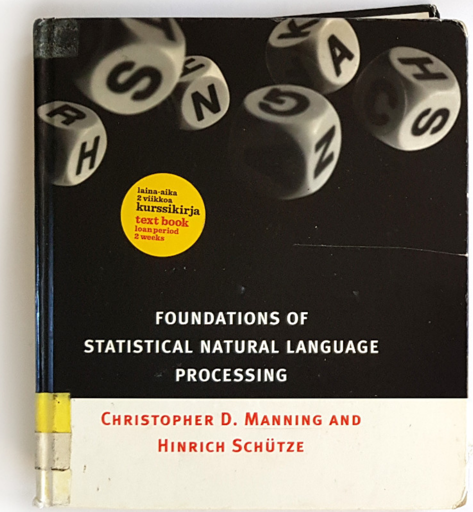
- **The course lasts through both period I and period II.**
- Lectures on Tuesdays 14-16 each week in Zoom: links to the Zoom meetings will be given in the course Moodle.
- Lecture recordings will be available in Moodle using the Panopto system
- On-site participation available for some lectures, lecture room: centre campus, main building, lecture hall D11. More information later.
- **No exercise sessions**, instead home exercise packs, see next slide.
- Language: English
- 5 ECTS credits

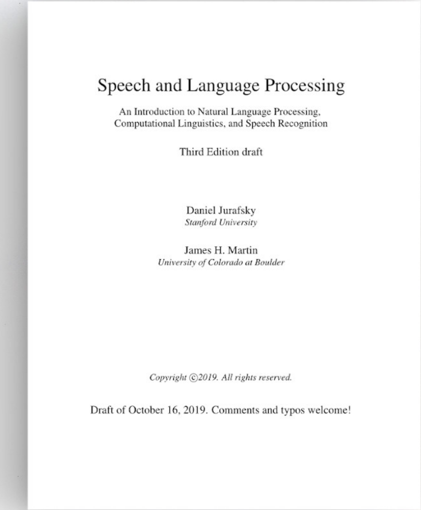
Course material

- Course material is provided through the Moodle: https://moodle.tuni.fi/course/view.php?id=27708
- Material:
  - Zoom lectures and their recordings
  - course slides
  - additional-reading articles (only if specified)
  - Exercise packs released weekly during the fall.
- Exercise packs will contain some mathematical exercises, some implementation & testing of methods, either from scratch or using pre-existing toolboxes.

Books

- The course is partly based on
  - C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing,* The MIT Press, 1999.
  - I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, The MIT Press, 2016
  - D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing,Computational Linguistics, and Speech Recognition.* 3rd edition, draft available at http://web.stanford.edu/~jurafsky/slp3/
  - S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit*. Available at https://www.nltk.org/book/
  - Various scientific papers




Speech and Language Processing

An Introduction to Natural Language Processing,
Computational Linguistics, and Speech Recognition

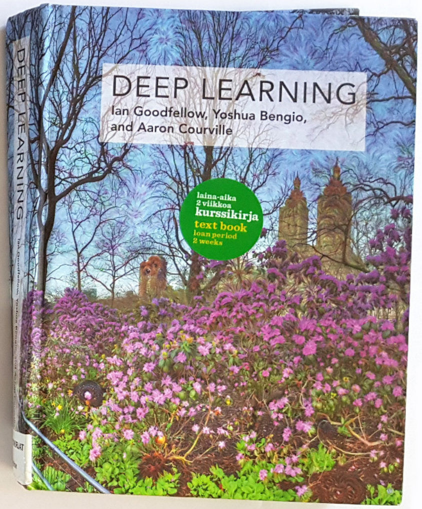
Third Edition draft

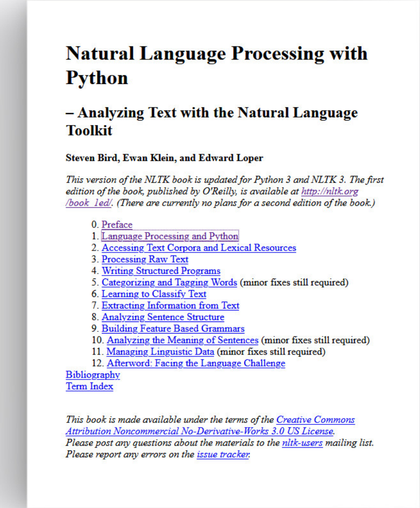
Daniel Jurafsky
Stanford University

James H. Martin
University of Colorado at Boulder

Copyright ©2019. All rights reserved.

Draft of October 16, 2019. Comments and typos welcome!




Natural Language Processing with Python

– Analyzing Text with the Natural Language Toolkit

Steven Bird, Ewan Klein, and Edward Loper

This version of the NLTK book is updated for Python 3 and NLTK 3. The first edition of the book, published by O'Reilly, is available at http://nltk.org/book_1ed/. (There are currently no plans for a second edition of the book.)

0. Preface
1. Language Processing and Python
2. Accessing Text Corpora and Lexical Resources
3. Processing Raw Text
4. Writing Structured Programs
5. Categorizing and Tagging Words (minor fixes still required)
6. Learning to Classify Text
7. Extracting Information from Text
8. Analyzing Sentence Structure
9. Building Feature Based Grammars
10. Analyzing the Meaning of Sentences (minor fixes still required)
11. Managing Linguistic Data (minor fixes still required)
12. Afterword: Facing the Language Challenge

Bibliography
Term Index

This book is made available under the terms of the Creative Commons Attribution Noncommercial No-Derivative-Works 3.0 US License.
Please post any questions about the materials to the nltk-users mailing list.
Please report any errors on the issue tracker.

- The books are not required reading: the slides are enough.

Deadlines

- Exam:
  - The first exam will be arranged after the final lecture. Retake exams will be given later.
- Deadlines:
  - Each weekly exercise set typically has 3 weeks time to complete the exercises. More time may be given for some weekly sets.
  - Late returns may be rejected or penalized in grading.
  - Answers for each weekly set must be returned in the corresponding return area in the Moodle.

Grading

- Each exercise graded 0-2 (integer), exercise packs total graded 0-5.
- Exam graded 0-5.
- To pass the course, you must pass the **exam** (grade 1 or more) and pass **exercise packs** (grade 1 or more).
- Passing grades are kept fractional between 1 and 5 (e.g. ”3.437”)
- Final course grade
  = round(0.8 * ExamGrade + 0.2 * ExercisesGrade)
  (e.g. 3.499 rounds to 3, 3.501 rounds to 4)

This course & speech recognition

- This course focuses on statistical methods, there is another course on rule-based methods:
  COMP.CS.360 Rule-based Natural Language Processing
- Language modeling is an important component of speech recognition: e.g., it helps recognize the next word based on what the language model thinks are likely continuations of previous recognized words.
- Because speech recognition has a lot of aspects of signal processing and acoustic (audio) modeling that go beyond text modeling, we do not deal with it on this course
- For more coverage of speech recognition, see for example the Tampere University courses
  - COMP.SGN.120 Introduction to Audio Processing, 5 ECTS
  - COMP.SGN.220 Advanced Audio Processing, 5 ECTS

This course & information retrieval

- Language modeling is an important component of information retrieval: it helps recognize the intent in search query phrases, recognize information in text documents, and match them.
- We will discuss information retrieval, but it has a lot of details that cannot be covered on this single course.
- For more coverage of information retrieval, see for example the Tampere University courses
  - ITAA02 Information retrieval, 5 ECTS
  - ITMS10 Information retrieval methods, 5 ECTS
  - ITMS11 Information retrieval and language technology, 5 ECTS
  - ITMS16 Task-based and interactive information retrieval, 5 ECTS

This course & deep learning

- Several recent language models are based on deep learning (deep neural networks).
- We will discuss selected deep learning based neural language models, but a full coverage of all aspects of deep learning more generally cannot be covered on this single course.
- For more coverage of deep learning, see for example the Tampere University courses
  - DATA.ML.200 Pattern Recognition and Machine Learning, 5 ECTS
  - DATA.ML.350 Neurocomputing, 5 ECTS

Contact Information

- Lecturer: Prof. Jaakko Peltonen, jaakko.peltonen@tuni.fi
- Office hours: room B0023 of the Pinni-B-building, by appointment only.
- When contacting by email, please include “DATA.STAT.840” in the subject line.
- Some questions can also be asked in the course discussion area in the Moodle

INTRODUCTION

Chapter 2: Perspectives

Perspectives on text analysis

- “Never trust the translation or interpretation of something without first trusting its interpreter. One word absent from a sentence can drastically change the true intended meaning of the entire sentence. For instance, if the word 'love' is intentionally or accidentally replaced with 'hate' in a sentence, its effect could trigger a war or false dogma.”
  — Suzy Kassem, *Rise Up and Salute the Sun*
- “If there really is such a thing as turning in one's grave, Shakespeare must get a lot of exercise.”
  — George Orwell, *All Art is Propaganda: Critical Essays*
  - Interpretation can (intentionally or unintentionally) add unintended meaning to text
- "If you give me six lines written by the hand of the most honest of men, I will find something in them which will hang him."
  — attributed to Cardinal Richelieu (disputed)
  - Interpretation can be actively biased, adversarial, or malicious

Perspectives on text analysis

- "Writing was born as the maidservant of human consciousness, but it is increasingly becoming its master. Our computers have trouble understanding how Homo sapiens talks, feels and dreams. So we are teaching Homo sapiens to talk, feel and dream in the language of numbers, which can be understood by computers."
  — Yuval Noah Harari, *Sapiens: A Brief History of Humankind*
    - Do we end up "dumbing down" the way we express ourselves to make it understandable to computers?
    - Think of search engine query phrases: very few write them in natural language!
    - And think of search engine optimization of websites: adding extra tags to content to make it easier for search engines to find.
- “As one Google Translate engineer put it, "when you go from 10,000 training examples to 10 billion training examples, it all starts to work. Data trumps everything.”
  — Garry Kasparov, *Deep Thinking: Where Machine Intelligence Ends and Human Creativity Begins*
    - Is this really true, or does it "just work" only for the common cases but still fail for the long tail of different uncommon cases?

Types of text data

Text, and need for analysis of text data, occurs in multiple domains:

- Literature: fiction, nonfiction in multiple genres. Online and digitized
- News: online news, digitized newspapers
- News comments
- Text content of webpages
- Search result snippets
- Online product descriptions
- Reviews: online and digitized
- Questionnaire answers
- Scripts and closed-caption tracks of movies and TV

Types of text data

- Scripts, closed-caption tracks, and other transcripts of online video
- Social media discussion
- Question-answer sites
- Instructions (e.g. recipes, manuals, online how-tos)
- Online and digitized encyclopedias
- Online and digitized textbooks
- Scientific research articles
- Textual annotations for various data (e.g. biological experiment databased; RDF databases)

Types of text data

- Laws
- Meeting transcripts (e.g. parliamentary records)
- Court case records
- Patents
- Customer service records
- Records of public service interactions: e.g. hospital patient records

English

- Text analysis and natural language processing methods have been developed for multiple languages.
- Usually at least the early stages of text processing require language-specific software, methods, and resources.
- Basic approaches and processing stages are shared between most languages, and some entire methods may work well enough for multiple languages.
- We focus on English.

Basic English grammar

- Parts of speech
  - nouns (common nouns *house, cat, computer*; proper nouns *Jaakko, Tampere, Finland*),
  - determiners (*the, a*)
  - pronouns (*he, she, it, we, they*),
  - adjectives (*yellow, happy, warm, difficult*),
  - verbs (*give, do, take*),
  - adverbs (*almost, only, happily*)
  - prepositions (*of, in, to, from, by, between*)
  - conjunctions (*and, or, but*)
- Noun phrases: *that quite easy exercise problem where I used Matlab*
- Verb forms: infinitive, tenses (present, past, continuous, perfect, perfect continuous, future, conditionals); passive voice; verb phrases/predicates
- Adjective forms: comparative, superlative; irregular forms; adjective phrases (*more than a little concerned*)
- Pronoun cases (nominative/subjective, oblique, genitive, reflexive: *I, me, my/mine, myself*)
- Declension (singular, plural, possessive: house, houses, house's, houses')
- Negation (not, don't, can't, shouldn't, never, nobody, nothing)
- Word order (usually subject-verb-object but sometimes object-verb-subject; question and imperative forms (*You can learn it. Can you learn it? Learn it!*)
- Clauses and sentence structure: independent clause, dependent clauses (dependent words, content clause, relative clause - Whereever that driver who just passed us is going, it seems he is in a hurry); subject-predicate-object; punctuation

Language and grammar changes

- It is not possible to cleanly delineate "correct" from incorrect language
- Spelling, grammar and word meanings change over time, new words and phrases get introduced
- British vs American English (*modeling* vs *modelling*)
- Internet language (*lol, o rly, k, brb noob, awk roflmao*)
- *fantastic* used to mean imagined things
- *nice* used to mean foolish/ignorant, then (14th-15th century) shy/reserved, then refined/polite
- *meat* used to mean any solid food, then (14th-15th century) animal flesh, then (20th century) also central part, *meat of the matter*
- *guy* used to mean Guy Fawkes effigies

Meaning is far more than syntax

- Topical content
- Facts, opinions, interpretations
- Narrative
- References
- Replies
- Sentiment
- World knowledge
- Aim of the text (providing information, telling a story, asking for information, arguing in favor/against something, influencing/persuading someone, critique, reviewing, making arrangements, ...)

Does being able to process natural language imply intelligence?

- "The ability to speak does not make you intelligent."

  — Qui-Gon Jinn to Jar-Jar Binks, Star Wars
  Episode I: The Phantom Menace

- However, several tests and challenges related to artificial intelligence have been proposed based on processing and generation of natural language

Turing test

- Influential concept in philosophy of artificial intelligence:
  - Human evaluator judges natural-language conversations between two participants.
  - The evaluator knows one of them is human and the other one is a machine, but does not know which one is which.
  - If the evaluator cannot accurately (reliably) tell which one is which, the machine passes the Turing test.
- Turing's 1950 article had two "imitation game" versions:
  - two human participants (A = a man and B = a woman) both try to convince the evaluator that they are the woman.
  - Version 1: A is replaced by a machine. Will the evaluator decide incorrectly as often as when A was human?
  - Version 2: A replaced by a machine, B by a man, both try to trick the evaluator into believing they are the woman.

Chinese room argument

- Thought experiment by John Searle (1980)
- Suppose a computer and program can take Chinese characters as input and produce Chinese characters as output, convincingly enough to convince a human Chinese speaker the program is a Chinese-speaking human and thus pass the Turing test.
- Does the computer understand Chinese, or just simulate ability to understand it?
- Suppose a non-Chinese-speaking human is in a room with Chinese phrasebooks; upon receiving a question in Chinese the human looks up that character sequence in the phrasebooks and responds with the given answer from the phrasebook. If the computer would pass the Turing test, would this "running the program manually" pass too?

Loebner prize

- Annual competition, launched in 1990, for AIs that converse with humans and responds to human questions (**chatterbots**).
- Prize for most human-seeming AI
- Never awarded: 1) prize for AI that judges cannot tell from a human; 2) another one that includes non-text inputs too.
- Latest competitions won by *Mitsuku* chatterbot (www.mitsuku.com), 5 total wins so far (2013, 2016, 2017, 2018, 2019)

A conversation with Mitsuku (20.7.2020)

- Hi Mitsuku. What would you say is your greatest flaw as a chatterbot?
- That was my previous name. My new name is Kuki. I don't have a favourite flaw as a chatbot. My favourite colour is blue.
- What do you like about that color?
- I can't really narrow it down to just one thing. I like everything which is why it is my favourite.
- Is there something you do not like about it?
- Not at the present time.
- What would you use the color for?
- in support of; in favor of
- In support of what?
- What of it? You sound like a category C client.
- What does category C mean?
- C stands for "critical", usually highly-educated clients who often have unsatisfying conversations with me.
- Unsatisfying in what sense?
- not up to expectations
- What kinds of expectations are they?
- Does it have anything to do with https://www.psychologytoday.com/us/blog/cui-bono/201802/the-psychology-expectations. ?
(The linked article is The Psychology of Expectations: Why unrealistic expectations are premeditated resentments.)
- What is that article about?
- It is about science fiction.
- Is it a story?
- yes. Do you like stories?
- Sometimes. What is the story about?
- When? Yesterday, Kuki was feeling bored so she decided to go out for a while. On her way there she met her friend, Alice and they decided to go to the river instead. While they were walking, Kuki hopped and skipped which made her feel a lot happier. The End
- Is that the story in the article?
- "The End"? Is it the story in the article?
- No, the entire thing, is that the story?
- How come? What is it about?
- I'm talking about the story you just told me.
- Yes, it is the topic.
- Topic of what?
- What of it? You sound like a category C client.
- Is a category C client the topic of the story?
- A category is the basic unit of knowledge in AIML.
- How does AIML relate to the story?
- It's the language I use to understand what you humans say to me.
- How does it work?
- AIML consists of a list of statements called categories. Each category contains an input pattern and a reply template. The syntax of an AIML category is:
<category>
<pattern> PATTERN </pattern> <template> Template </template>
</category>
or
<category>
<pattern> PATTERN </pattern>
<that> THAT </that>
<template> Template </template>
</category>
The AIML category tags are case-sensitive. Each open tag has an associated closing tag. This syntax obviously derives from XML.

GPT-2 and GPT-3

- GPT-2 and its successor GPT-3 are language generation models by the company OpenAI
- Technology review 14.2.2019 about GPT-2: "An AI that writes convincing prose risks mass-producing fake news": https://www.technologyreview.com/2019/02/14/137426/an-ai-tool-auto-generates-fake-news-bogus-tweets-and-plenty-of-gibberish/
- GPT-2 has been released: https://github.com/openai/gpt-2
- GPT-3 has not been released in full yet (some data is at https://github.com/openai/gpt-3), but some experiments have been done with an API: e.g. https://maraoz.com/2020/07/18/openai-gpt3/

Hutter prize

- Cash prize for advancement in compression of a block of Wikipedia text: 5000 euros for each percent of improvement.
- Organizers believe progress towards text compression leads to progress towards artificial intelligence
- Current achievement: 1Gb text corpus "enwik9" compressed to about 115Mb (winner: Artemiy Margaritov, May 13, 2021)
- Previous smaller task: 100Mb text corpus "enwik8" compressed to about 15Mb (winner: Alexander Rhatushnyak, November 4, 2017)

INTRODUCTION

Chapter 3: Preliminaries

Some basic rules of probability

- Sum/integral over the set of possibilities is 1:

$$\sum_k p(x=k) = 1 \qquad \int_y p(y)=1$$

- Marginal probability:

$$\sum_k p(x=k,y) = p(y)$$

- Conditional probability:

$$p(y|x) = \frac{p(y,x)}{p(x)}$$

- Chain rule of conditional probabilities:

$$p(a,b,c,\ldots,y,z)=p(a|b,c,\ldots,y,z)\,p(b|c,\ldots,y,z)\cdots p(y|z)\,p(z)$$

- Bayes rule: $p(y|x)=\dfrac{p(y,x)}{p(x)}=\dfrac{p(x|y)\,p(y)}{p(x)}$

Some basic machine learning concepts

- **Data set** of **observations**, each observation has one or more **features.**
- Often data is divided into a **training set** and **test set** (**cross-validation** uses several divisions)
- Observations may form a (time) **series** (e.g. consecutive words in a document)
- Observations may be divided into **subsets** (several paragraphs, documents, document sources, social media conversation threads and replies)
- Tasks may be **supervised** (try to imitate known solution examples) or **unsupervised** (try to find interesting things)
- Typical supervised tasks: **regression**, **classification**
- Typical unsupervised tasks: **clustering, extracting components, visualization**

Some basic machine learning concepts

- **Performance measures:** evaluate quality of a model.
  - In supervised tasks, usually compare model outputs to correct results.
    - **Mean squared error** in regression: compare prediction $\hat{y}$ to correct value y , compute $\left(y-\hat{y}\right)^2$, take average over samples
    - **Classification error**: compare predicted class $\hat{c}$ to correct class c , add 1 error if they are not the same, sum over samples
  - In unsupervised tasks, usually some mathematical criterion that does not require known correct results.
    - **Pairwise distance** in clustering: take all pairs $x_{1,} x_2$ of observations in the same cluster, compute distance $\|x_1-x_2\|^2$, sum over all pairs per cluster and over all clusters
    - **Probability of observations (likelihood**)
    - **Posterior probability of parameters**

Some basic machine learning concepts

- **Optimization: improve** quality of a model (improve the value of a performance measure) by changing its parameters.
  - **Gradient descent**: compute the derivative of the performance measure L for each parameter u: $\frac{\partial L}{\partial u}$.
    To grow L, change u by $u_{new} = u + \alpha \cdot \frac{\partial L}{\partial u}$ (α= learning rate)
    To decrease L, change u by $u_{new} = u - \alpha \cdot \frac{\partial L}{\partial u}$
    Repeat many times.
  - In some cases it is possible to compute the optimal parameter value (e.g. zero-point of the derivative) directly, e.g. convex optimization
  - Other algorithms include e.g. **Newton's method**, **conjugate gradient** (searches in 'conjugate' directions to avoid redundancy), **stochastic gradient** (replaces long sums by sampled indices), **L-BFGS** (limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm) and many others

Some basic machine learning concepts

- **Optimization for probabilistic models:** typically either **maximum likelihood**, **maximum a posteriori**, or **posterior sampling**
  - **Maximum likelihood**: find the parameter values that make the probability of observed data as high as possible.

    $$\theta_{ML} = max_{\theta}\, p\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T | \theta\right)$$

    Can be done by the optimization algorithms on the previous slide.
  - **Maximum a posteriori**: find the parameter values that maximize the posterior probability of parameters given data:

    $$\theta_{ML} = max_{\theta}\, p\left(\theta | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\right) = max_{\theta}\, p\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T | \theta\right) p\left(\theta\right)$$

  - **Posterior sampling**: create a set of samples distributed according to the posterior probability of parameters given data $p\left(\theta | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\right)$.
    - Typically, samples are generated iteratively as chains where the generation of the next sample depends on the previous one.
    - Algorithms include **Metropolis-Hastings** and **Gibbs sampling**

Software

- Natural language processing tools are available in several programming languages
- **Python**:
  - natural language toolkit (NLTK): www.nltk.org
  - **We will use this on the course.**
- **R**: packages tm, openNLP, RWeka, tidytext, quanteda, text2vec and others: https://cran.r-project.org/web/views/NaturalLanguageProcessing.html
- **Matlab and Octave**:
  - Text Analytics Toolbox
  - MatlabNLP: https://github.com/faridani/MatlabNLP

Software

- Natural language processing tools are available in several programming languages
- **Java**:
  - Stanford CoreNLP: https://stanfordnlp.github.io/CoreNLP/
  - Apache OpenNLP: https://opennlp.apache.org/
  - Apache Lucene (information retrieval library): https://lucene.apache.org/core/
  - GATE ("General Architecture for Text Engineering"): https://gate.ac.uk/
  - MALLET ("MAchine Learning for LanguagE Toolkit"): http://mallet.cs.umass.edu/
- **C++:**
  - MeTA: https://meta-toolkit.org/
- **Ruby:**
  - Stanford CoreNLP: https://github.com/louismullie/stanford-core-nlp
  - OpenNLP: https://github.com/louismullie/open-nlp

Software

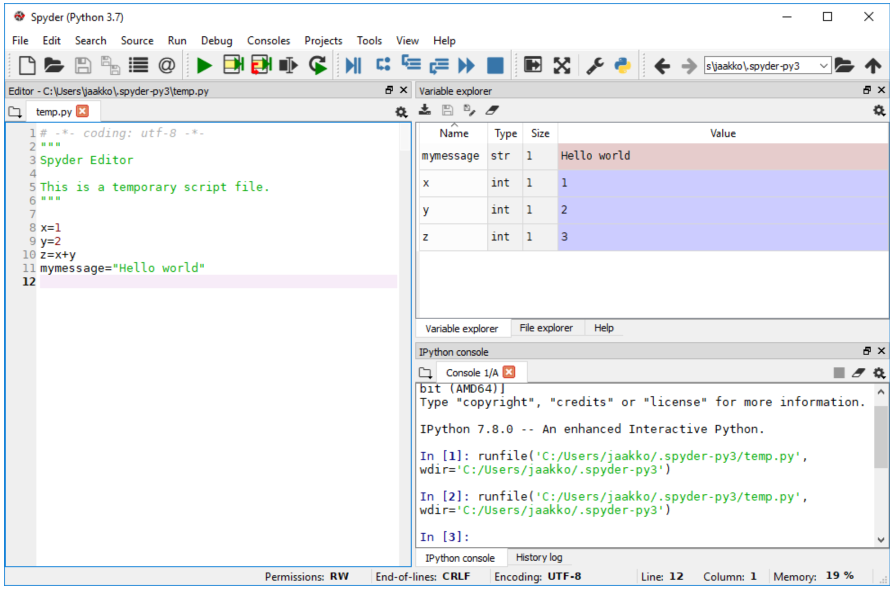
- Natural language processing tools are available in several programming languages
- **Javascript and node.js**:
  - nlp.js: https://github.com/axa-group/nlp.js
  - Natural: https://github.com/NaturalNode/natural
  - compromise.cool: https://github.com/spencermountain/compromise/
  - Wink NLP utils: https://github.com/winkjs/wink-nlp-utils
- **C#:**
  - SharpNLP: https://archive.codeplex.com/?p=sharpnlp
- **PHP**:
  - NLP-tools: http://php-nlp-tools.com/
  - nlpTools: http://nlptools.atrilla.net/web/
  - PHP Text Analysis: https://github.com/yooper/php-text-analysis

Software

- Various paid **cloud processing** solutions are available (these are just some of many companies offering such services):
  - Google Cloud Natural Language (C#, Java, Go, node.js, PHP, Python, Ruby): https://cloud.google.com/natural-language/docs/quickstart-client-libraries
  - Microsoft Azure Cognitive Services, Language Understanding (https://www.luis.ai/, https://docs.microsoft.com/en-us/azure/cognitive-services/luis/) and Text Analytics (https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/)
  - IBM Cloud, Watson Natural Language Understanding (https://www.ibm.com/fi-en/marketplace/natural-language-understanding, https://cloud.ibm.com/apidocs/natural-language-understanding)
  - Amazon, Amazon Comprehend (https://aws.amazon.com/comprehend/) and Amazon SageMaker (https://aws.amazon.com/sagemaker/)

Installing Python and NLTK

- We use Python version 3.8 or above.
- I recommend the Anaconda distribution, https://www.anaconda.com/distribution/
and the Spyder development environment
(part of the Anaconda distribution).
Installers are available for Windows,
MacOs and Linux.


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
s\jaakko\.spyder-py3
Editor - C:\Users\jaakko\.spyder-py3\temp.py
temp.py
Variable explorer
Variable explorer File explorer Help
IPython console
Console 1/A
IPython console History log
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 12 Column: 1 Memory: 19 %

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""

x=1
y=2
z=x+y
mymessage="Hello world"
```

Name	Type	Size	Value
mymessage	str	1	Hello world
x	int	1	1
y	int	1	2
z	int	1	3

```
bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/jaakko/.spyder-py3/temp.py',
wdir='C:/Users/jaakko/.spyder-py3')

In [2]: runfile('C:/Users/jaakko/.spyder-py3/temp.py',
wdir='C:/Users/jaakko/.spyder-py3')

In [3]:
```

- Anaconda has NLTK and numpy preinstalled. For other distributions, you can install NLTK and numpy:
  - Linux/MacOs:
    ```
    pip install --user -U nltk
    pip install --user -U numpy
    ```
  - Windows: after installing Python, install numpy from https://www.scipy.org/scipylib/download.html and NLTK from https://pypi.org/project/nltk/
- To use NLTK, numpy or any other Python library, import it in the Python console:
  ```
  import numpy
  import nltk
  ```
- Within NLTK, sometimes you need to install resources (modules or data) using the NLTK downloader, e.g.:
  ```
  nltk.download('punkt')
  ```

Python basics

- The python interpreter can perform basic calculations:

```
>>> 1+2
Out[142]: 3
```

- Parentheses, plus, miinus, division:

```
>>> 1+2*(3-4)/26
Out[143]: 0.9230769230769231
```

- Remainder with the % symbol:

```
>>>  3 % 2
Out[146]: 1
>>>  3.1 % 1.7
Out[147]: 1.4000000000000001
```

- Power with two asterisks: **

```
>>>  3 ** 2
Out[148]: 9
```

- Note: ^ **is not power (it is a bitwise exclusive-or)**

Python basics

- Variable definition: type assigned by value

```
x=1;                               # Integer
z=3.54;                            # Float
mymessage='Hello world';           # String, note \'  \"  \\  \n
assumption1=True;                  # Boolean
assumption2=False;                 # Boolean
```

- Case matters, **`y`** is a different variable than **`Y`**
- A created variable can be removed with the del command:

```
x=1;
del x;
x
Traceback (most recent call last):
  File "<ipython-input-80-401b30e3b8b5>", line 1, in <module>
    x
NameError: name 'x' is not defined
```

Python help functions

- List variables, builtin functions, and imported modules seen by the interpreter:

```
dir();   # list variables and functions
```

- List functions offered by a library:

```
dir(math);# list functions & attributes of the module 'math'
```

- Help for functions:

```
help(print);   # describe the print function
help(math.cos); # describe the cos function of the math module
```

- Check the type of a variable

```
type(x); # tells whether x is an integer, float, string etc.
```

The variable explorer in the Spyder interface also shows types and values of the variables seen by the interpreter

Python basics: Conditionals

- Conditional statements are indented:

```
if x<10:              #  Condition
    x=x+10;           #  Happens if the condition is true
    x=x*2;            #  Happens if the condition is true
print(x);             #  Happens after the conditional
```

- Different conditional clauses:

```
if x>10:              #  Greater than
if x>=10:             #  Equal or greater than
if x<10:              #  Smaller than
if x<=10:             #  Equal or smaller than
if x==10:             #  Equal to
if x!=10:             #  Not equal to
if (x<10)|(x>20):          #  OR: either condition must hold
if (x<10)&(x*x<5):         #  AND: both conditions must hold
```

Python basics: Repetition statements

- Statements in a repetition block are indented:

```
x=0;
for i in range(10):          # range(10) means numbers 0-9
    x=x+i;                   # Happens inside the loop
print(x);                    # Happens after the loop

x=0;
while x<10:                  # range(10) means numbers  0-9
    x=x+i;                   # Happens inside the loop
print(x);                    # Happens after the loop
```

- You can't perform computations with a range command, e.g. range(10)+1 does not work.
  - Solution: transform it into an **array** before computations.
- Statements **continue** and **break** can be used to skip an iteration or exit from the loop

Python basics: Data structures

- Tuple:

```
mytuple = (1, 2, 3, 'apple', 13.5)
mytuple[3]                    # 'apple'
len(mytuple)                  # 5 elements
```

- List:

```
mylist = [1, 2, 3, 'apple', 13.5, [11,12,13]]
mylist[0] = mylist[0]+10;     # 1+10=11
mylist[3] = mylist[3]+' tea'; # 'apple tea'
```

- Dictionary (name-value pairs):

```
movie = {'name':'A Beautiful Mind','director':'Ron
Howard','year':2001}
movie['name']    # 'A Beautiful Mind'
```

- Array (like a multidimensional list):

```
import numpy;
A=numpy.array([[1,2,3],[4,5,6]]);
C=numpy.array([['coffee',2],['tea',5]]);
A[1,2]    # 6
C[1,0]    # 'tee'
```

Python: some libraries to be aware of

- **Packages in red are not part of the anaconda distribution, must be installed separately**
- **os**: changing directory paths
- **runpy**: running code from files
- **pickle**, **shelve**: saving variables to files

- **math**: basic mathematical functions
- **cmath**: handling complex numbers
- **statistics**: statistical functions
- **random**: random number generation
- **numpy**: handling arrays and basic calculations
- **scipy**: advanced calculation

Python: some libraries to be aware of

- **matplotlib**: making plots.
- **seaborn**, **bokeh**, **plotly**, **pydot**: other plotting libraries
- **csv**: .csv file import
- **scrapy**, **beautifulsoup (bs4)**: fetching data from websites
- **string**: handling strings
- **time, calendar, datetime**: handling dates and times
- **gzip**, **zipfile**, **tarfile**: handling compressed files

Python: some libraries to be aware of

- **sympy**: symbolic calculus
- **pandas**: working with data files
- **keras**, **pytorch**, **tensorflow**: neural network and deep learning libraries

Python resources online

- Python documentation: https://docs.python.org/
- Python's own tutorial: https://docs.python.org/3/tutorial/
- Python wiki beginner's guide: https://wiki.python.org/moin/BeginnersGuide
- Wikibooks guide: https://en.wikibooks.org/wiki/Python_Programming
- Numpy, scipy, matplotlib, etc. documentation: https://scipy.org/docs.html

- Many others! For example,
  LearnPython tutorial: https://www.learnpython.org/

  The Hitchhiker's Guide to Python tutorial: https://docs.python-guide.org/intro/learning/

Python books

- Several Python books are available for purchase:

Thomas Haslwanter
An Introduction to Statistics with Python
With Applications in the Life Sciences
Springer
José Unpingco
Python for Probability, Statistics, and Machine Learning
Second Edition
Springer
Comparative Approaches to Using R and Python for Statistical Data Analysis
Prabhanjan Narayanachar Tattar
Statistical Application Development with R and Python
Second Edition
Power of Statistics using R and Python
Packt
DOING MATH WITH PYTHON
AMIT SAHA
O'REILLY
Python Data Science Handbook
ESSENTIAL TOOLS FOR WORKING WITH DATA
Laura Igual Santi Seguí
Introduction to Data Science
A Python Approach to Concepts, Techniques and Applications
Springer
DATA SCIENCE AND ANALYTICS WITH PYTHON
Jesús Rogel-Salazar
CRC Press
Python Data Analytics
With Pandas, NumPy, and Matplotlib
Second Edition
Fabio Nelli
Apress
PYTHON DATA SCIENCE
DATA SCIENCE USING PYTHON AND R
CHANTAL D. LAROSE | DANIEL T. LAROSE
WILEY
Python for Data Science dummies
John Paul Mueller
Luca Massaron
O'REILLY
Introduction to Machine Learning with Python
Andreas C. Müller & Sarah Guido
Learn By Examples
A Quick Guide To Data Science With Python
INTRODUCTION TO Computational Models with Python
José M. Garrido
Hands-On Data Analysis with NumPy and pandas
Packt
INTRODUCTION TO MODELING AND SIMULATION WITH MATLAB® AND PYTHON
Steven I. Gordon and Brian Guilfoos
Python Machine Learning
Sebastian Raschka
Packt
Frank Kane
Hands-On Data Science and Python Machine Learning
Packt
Training Systems Using Python Statistical Modeling
Packt
Data Science Essentials in Python
Collect
Organize
Explore
Predict
Value
DEEP LEARNING with Python
François Chollet
MANNING
Bayesian Analysis with Python
Second Edition
Packt
Python Data Science Essentials
Second Edition
Packt
O'REILLY
Python for Data Analysis
Wes McKinney
Introduction to Python for Econometrics, Statistics and Data Analysis