The ABC of Computational Text Analysis

#9 Introduction to Python

Alex Flückiger
29 April 2021

Faculty of Humanities and Social Sciences
University of Lucerne

Recap last Lecture

- from words to embeddings 🌈

  relational, re-contextualized word meaning

- data-driven NLP is both powerful and biased 👊 🤕
- data is never raw but depends on many decisions 💭
- Questions ❓

Outline

enter the shiny world of Python 😎

- development environment
- basic Python syntax


Python

Python is ...

a programming language

- general-purpose

  not specific to any domain

- interpreted

  no compiling

- standard language in data science

How to learn programming?

three inconvenient truths 😰

- programming cannot be learnt in a course

  *my goal is to make the **start** as easy as possible!*

- frustration is normal

  ***fight** your way!*

- the Python ecosystem is huge

  ***grow** skills by step-by-step!*

Programming can be absolutely captivating! ✌️

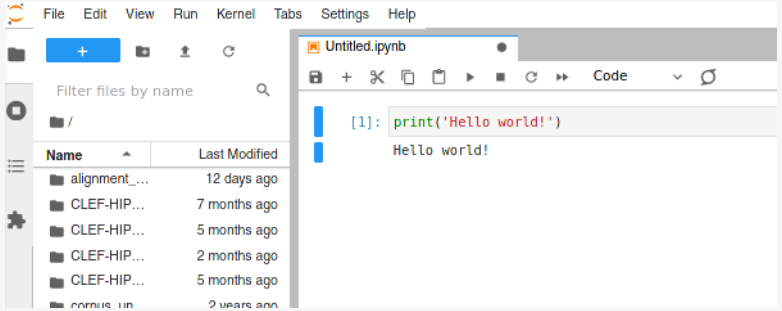
Jupyter Lab Editor

Getting started

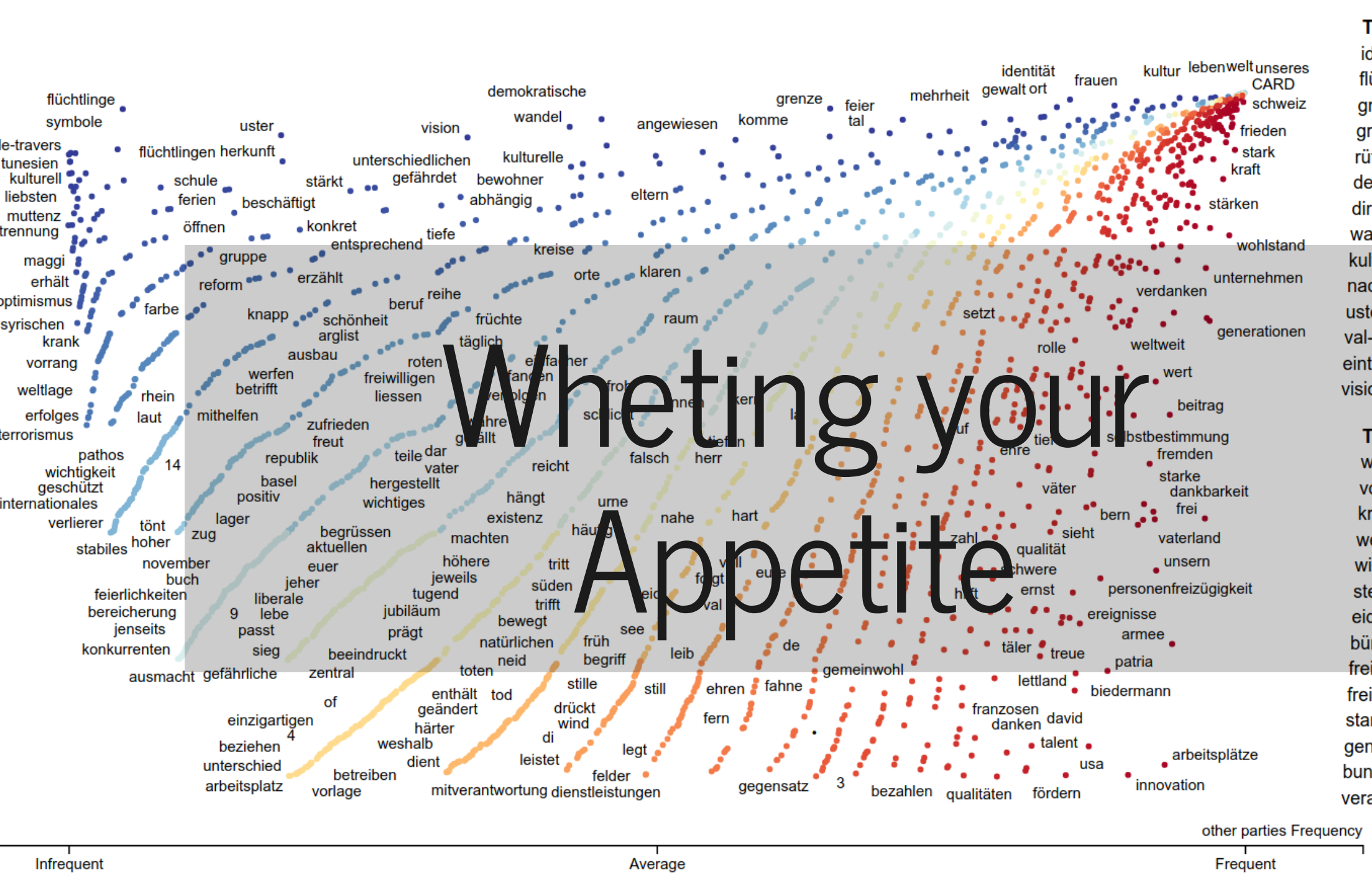
- Open Terminal/Ubuntu
- `jupyter lab --no-browser`
- Copy the link and open in your browser
- `print("Hello, World!")`

Features

- web-based editor
- interactive development
- showing code next to output and comments
- language-agnostic



Jupyter Lab Editor


Wheting your Appetite
Top SP
identität
flüchtlinge
grenze
grenzen
rütli
demokratische
direkte
wandel
kultur
nachbarn
uster
val-de-travers
eint
vision
Characteris
CARD
unseres
freiheit
demokratie
eidgenossen
wohlstand
zukunft
unserem
eidgenossen
unseren
verantwortung
gemeinsam
politischen
bundesrat
nationalfeierta
Top other parties
wohlstand
volk
kraft
werte
wirtschaft
steht
eidgenossen
bürger
freiheit
frei
stark
generationen
bundesbrief
verantwortung
vertrauen
werte
schweiz
vielfalt
bewusst
wirtschaftlichen
zusammenhalt
feiern
generationen
bundesbrief
menschen
schliesslich
schweizerinnen
heutigen
kantone
other parties Frequency
Infrequent
Average
Frequent

Python Syntax

Variables

variables are kind of storage boxes

```
# define variables
x = "at your service"
y = 2
z = ", most of the time."

# combine variables
int_combo = y * y       # for numbers any mathematical operation
str_combo = x + z       # for text only concatenation with +

# show content of variable
print(str_combo)
```

Data Types

The object's type is implicit (dynamic)

Name	What for?	Type	Examples
String	Text	str	`"Hi!"`
Integer, Float	Numbers	int, float	`20`, `4.5`
List	Lists (ordered, mutable)	list	`["Good", "Afternoon", "Everybody"]`
Boolean	Truth values	bool	`True`, `False`
⋮	⋮	⋮	⋮
Tuple	Lists (ordered, immutable)	tuple	`(1, 2)`
Dictionary	Relations (unordered, mutable)	dict	`{"a":1, "b": 2, "c": 3}`

Data Type Conversion

combine variables of the same type only

```
# check the type
type(YOUR_VARIABLE)

# convert types (similar for other types)
int('100')  # convert to integer
str(100)    # convert to string

# include variable in a f-string
x = 3
mixed = f"x has the value: {x}"
print(mixed)
```

Confusing Equal-Sign

= vs. == contradicts the intuition

```
# assign a value to a variable
x = 1
word = "Test"

# compare two values if they are identical
1 == 2              # False
word == "Test"      # True
```

Comments

- `# comments` ~ lines ignored by Python
- do it, it helps you …
  - *to learn initially*
  - *to understand later*

```
# single line comment

"""
comment across
multiple
lines
"""
```

Iterations

for-loop

do something with each element of a collection

```
sentence = ['This', 'is', 'a', 'sentence']

# iterate over each element
for token in sentence:

    # do something with the element
    print(token)
```

Conditionals

if-else statement

condition action on variable content

```
sentence = ['This', 'is', 'a', 'sentence']

if len(sentence) == 3:
    print('This sentence has exactly 3 tokens')
elif len(sentence) < 5:
    print('This sentence has less than 5 tokens')
else:
    print('This sentence is longer than 5 tokens')
```

Indentation

indentation matters!

- intend code within code blocks

  loops, if-statements etc.

- press `tab` to intend

✅

```
if 5 > 2:
    print('5 is greater than 2')
```

❌

```
if 5 > 2:
print('5 is greater than 2')
```

Methods

```
# split at whitespace
tokens = 'This is a sentence'.split(' ')

# check the variable
print(tokens, type(tokens))

# add something to a list
tokens.append('.')

# join elements to string
tokens = ' '.join(tokens)
print(tokens, type(tokens))
```

Functions and Arguments

- functions have the form

  `function_name(arg1, arg2)`

- functions may have arguments

```
# define a new function
def word_properties(word):
    """
    Print some properties of the provided word.
    It takes any string as argument (variable word).
    """

    # print(), len() and sorted() are functions themeselves
    word_length = len(word)
    sorted_letters = sorted(word, reverse=True)
    print(f"Properties for the word '{word}':")
    print("length:", word_length, "letters:", sorted_letters)

word_properties("computer") # call function with the word "computer"
```

Indexing

Python starts counting from zero!

```
sentence = ['This', 'is', 'a', 'sentence']

# element at position X
first_tok = sentence[0]     # 'This'

# elements of subsequence [start:end]
sub_seq = sentence[0:3]     # ['This', 'is', 'a']

# elements of subsequence backwards
sub_seq_back = sentence[-2:]        # ['a', 'sentence']
```

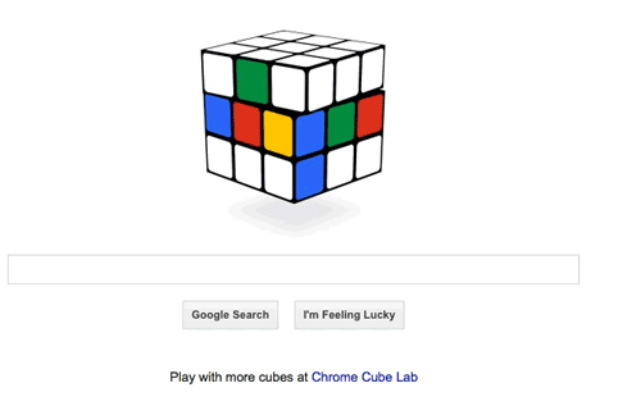
Syntax Errors

1. read the error message
2. find the source of the error

   script name + line number

3. paste message into Google

   check answers on stackoverflow



Learning by doing, doing by googling

Modules/Packages

- modules provide functionalities
- no programming from scratch 🎉

NLP Packages

- spaCy

  industrial-strength Natural Language Processing (NLP)

- textaCy

  NLP, before and after spaCy

- scattertext

  beautiful visualizations of how language differs across corpora

In-class: Install Packages for next week

open Terminal/Ubuntu

```
# install scattertext
conda install -c conda-forge scattertext

# install language specific models for spacy
python -m spacy download de_core_news_sm
python -m spacy download en_core_web_sm
```

In-class: Exercises I

1. Make sure that your local copy of the Github repository KED2021 is up-to-date with `git pull`. Open the Notebook with the Python Basics in Jupyter Lab: `materials/code/python_basics.ipynb`.
2. Try to understand and run the commands line-wise. Modify them to see how the output changes. Initially, the try-and-error is good strategy to learn.

In-class: Exercises II

1. Write a Python script in Jupyter Lab that

   takes a text (a string)
   splits it into words (a list)
   iterates over all the tokens and print all tokens that are longer than 5 characters
   Bonus: wrap your code in a function.

2. Go to the next slide. Start with some of the great interactive exercises out there in the web.

Resources

learn basics interactively

- Python Principles
- LearnPython

official Python introduction

- Python introduction


Questions?