Tutorial 1: Basic NLP Pipeline

The goal of the first tutorial is to create an NLP pipeline and get familiar with the common NLP modules. You will be able to handle most of the tasks with pandas as well as the nltk module, which are the basis for many NLP projects. Please use a (Jupyter) notebook for your code, for keeping things clean, performative and easy to annotate. Believe me, it's worth the extra effort!

We will NOT post "ideal solutions" to the tutorials, instead, we will "live code" a solution in the tutorial session and talk about the hows and whys. To get the most benefits from our sessions, be prepared and try to solve everything beforehand, good questions and your input will contribute significantly to the quality of the tutorials!

Aufgabe 1: Installation and Import

1.1: Installation and Import of NLP Modules

Make sure that the required modules "pandas", "numpy", "nltk", "sklearn", "spacy" and "re" are imported and get the version numbers of the modules.

Hint: Use `! pip install NAMEOFMODULE` to run command line / shell commands from your python jupyter notebook and install modules without having to use your terminal.

1.2 Import of "Quality-of-Life Modules"

Install "pandarallel". Import pandarallel's method "pandarallel" (for parallelization in pandas) and initialize it with pandarallel.initialize(). Consult pandarallel's documentation or google if you need help with using pandarallel.

2: Import Data

In this tutorial you will work with a dataset of jokes. These were collected using crawlers from the platforms "stupidstuff.org", "wocka.com" as well as "reddit.com".

Read in the attached .json files as pandas dataframes and merge them into a single dataframe. When doing so, make sure that the source of the data is preserved as a key.

3: Data Preprocessing

As is very often the case with crawled data, the data from the different sources are very differently sourced and tagged.

3.1: Clean up text

Prepare the data so that the "body" column contains the entire text of the joke and that no format tokens (e.g. "\n") are included.

3.2: Tokenization

Use the nltk module to tokenize the jokes. Use the RegexpTokenizer of the nltk module to take only tokens from words and numbers (no punctuation marks) with a suitable regular expression. The tokens should contain only small letters. Save the results in a column "tokens".

3.3 Stopword removal

Remove all English stopwords from the generated tokens.

3.4 POS Tagging

Determine POS tags for the tokenized texts and store them in a "pos" column.

3.5 Lemmatization

Lemmatize the tokens of the texts and store the specific lemmas in a column "lemmata".

BONUS: Consider the word forms of the tokens during the lemmatization.

3.6 Frequencies

Add in a new column "frequencies" the frequency distributions of the lemmatized tokens for each text.

4 Data Analysis

4.1 Overview of topics

For the jokes crawled by Stupidstuff and Wocker, output the categories as a list.

4.2 Overview of numerical values

Display the average ratings for the categories of Stupidstuff jokes, sorted in ascending order. Then, using Pandas, display an overview of descriptive statistics of Stupidstuff jokes, again grouped by category.

5 Bonus

Analyze or edit an aspect of your choice of the dataset. For example, for each of the platforms, you can calculate the percentage of jokes that contain a specific word or words of a specific type of word, calculate overall frequencies of the different sources or genres, or edit any other arbitrary question that can be implemented without importing additional modules.