Symbolische Programmiersprache 4

Intro to NLTK

1. NLTK
   - a python library with lots of tools to dealing with natural language
   - Corpus: (Plural: Corpura) a collection of linguistic data
2. Tokenization
   - process of breaking raw text into its building parts: words, phrases, symbols, or other meaningful elements called tokens.
   - Tokenizing raw text, or getting a list of tokens, is almost always the first step to any NLP task, such as part-of-speech tagging and named entity recognition.
     - Types: number of distinct words
     - Tokens: total number N of running words
   - code

     ```
     >>> sent = "Dr. Jones is the U.S. President's advisor."
     # 1.attempt using regular expressions, chop on space
     >>> sent.split(' ')
     ['Dr.', 'Jones', 'is', 'the', 'U.S.',  "President's", 'advisor.']
     ```

     - a good starting point, but for English there are a number of tricky cases(eg. the various uses of the apostrophe for possession and contractions)
   - usage of NLTK
     - code

       ```
       >>> import nltk
       >>> text = "Mr. K.'s cars are different colors."
       >>> tokens = nltk.word_tokenize(text)
       >>> tokens
       ['Mr.', 'K.', "'s", 'cars', 'are','different', 'colors', '.']
       >>> " ".join(tokens)
       "Mr. K. 's cars are different colors ."
       ```

     - NLTK comes with a bunch of tokenization possibilities
       - `word_tokenize(s)`
       - `wordpunct_tokenize(s)`
       - `WhitespaceTokenizer().tokenize(s)` and `SpaceTokenizer().tokenize(s)`
3. Word counting
   - Frequency: measure similarity between documents by computing the `overlap` of their token frequency tables
4. Document representing
   - First we have to answer two questions:
     - How to tokenize a text
       - using regular expressions
       - using NLTK tokenization options
     - How to create a word-count dictionary
       - using `for Loop` and `dictionary`
       - using NLTK
         - code

           ```
           >>> import nltk
           >>> doc = ['This', 'is', 'a', 'sample', 'text', ..., 'frequencies',
           ```

```
>>> mytext = nltk.Text(doc)
>>> mytext
<Text: This is a sample text that we want...>
>>> freq = nltk.FreqDist(mytext)
>>> freq
FreqDist({'a': 3, 'as': 3, '.': 3, 'This': 2, 'text': 2, 'we': 2,
        'be': 2, 'document': 2, 'can': 2, ',': 2, ...})
>>> freq.most_common(4)
[('a', 3), ('as', 3), ('.', 3), ('This', 2)]
```

5. Accessing Corpora in NLTK
   - 要创建一个 Text 对象，需要先进行 标记化(Tokenization)

```
import nltk
from nltk.text import Text
from nltk.tokenize import word_tokenize
# 示例文本
text = "Natural Language Processing with NLTK is powerful and easy to use."
# 进行分词（Tokenization）
tokens = word_tokenize(text)
# 创建 NLTK 的 Text 对象
text_obj = Text(tokens)
print(text_obj)  # 输出: <Text: Natural Language Processing with NLTK...>
```

6. Concordances
   - a list of all occurrences of a given word together with its context. Also known as keyword in context.(查找单词出现的上下文)

```
text_obj.concordance("NLTK")
#output:
#Displaying 1 of 1 matches:
#Natural Language Processing with NLTK is powerful and easy to use
```

   - Distributional similarity:
     - 不同的单词在类似的上下文环境中出现的情况。如果两个词在相似的上下文中出现，它们可能具有相似的语义或功能。
     - `Text.similar`: finds other words which appear in the same contexts as the specified word; list most similar words first.
     - `Test.common_contexts`:
       - 用于 查找两个或多个单词在相同上下文(context)中出现的情况。它可以帮助我们分析哪些单词 在类似的语境中被使用，进一步验证 分布式相似性
   - Lexical disperation plot
     - dispersion plot: to display the location of a word in a text. 一种可视化工具，用于展示特定单词在文本中的分布情况。它可以帮助我们：观察单词在文本中的出现位置和频率。发现某些术语在文本中的主题模式。分析文本结构，例如书籍章节中的关键词分布。
     - ```
       import nltk
       from nltk.book import *
       # 选择目标单词
       target_words = ["Elizabeth", "Darcy", "Jane", "Bingley", "Wickham"]
       # 生成词汇分布图
       text2.dispersion_plot(target_words)
       ```

7. Basic text statistics
   - Lexical Diversity: Type-token Ratio (TTR)
       ```

    - Higher TTR indicates that a text has a greater variety of word types and integrates a higher amount of information.

8. Brown Corpus Stats
    - for english
9. Lexicon or lexical resource
    - a collection of words and/or phrases along with **associated information (annotation)**, e.g., definitions, sentiment polarity, etc
    - often secondary to texts; usually they are created and enriched with the help of texts
10. Frequency Distribution
    - Differ based on the text they have been calculated on
    - May also differ based on other factors: e.g. categories of a text (genre, topic, author, etc.)
        - we can measure different frequency distributions for each category
            - conditional frequency distributions
                - Each frequency distribution is measured for a different condition (e.g. category of the text)
                - vs. Frequency distribution counts observable events
                - Conditional frequency distribution needs to pair each event with a condition `(condition, event)`