Due date: October 3rd, 2021

Goal: running text preprocessing pipeline in NLTK and proofreading results
目标：在 NLTK 中运行文本预处理管道并校对结果

Data: NLTK version of Reuter's text training/267

> INDONESIA UNLIKELY TO IMPORT PHILIPPINES COPRA
> Indonesia is unlikely to import copra from the Philippines in 1987 after importing 30,000 tonnes in 1986, the U.S. Embassy's annual agriculture report said. The report said the 31 pct devaluation of the Indonesian rupiah, an increase in import duties on copra and increases in the price of Philippines copra have reduced the margin between prices in the two countries. Indonesia's copra production is forecast at 1.32 mln tonnes in calendar 1987, up from 1.30 mln tonnes in 1986.

Overview: Do the following project in NLTK. Get the text using the NLTK corpus access commands, do not cut and paste from the assignment. **Develop a single script called PreProcess** that executes the following pipeline in NLTK, taking the file to be preprocessed as a parameter, PreProcess(training/267)
概述：在 NLTK 中执行以下项目。使用 NLTK 语料库访问命令获取文本，不要从赋值中剪切和粘贴。开发一个名为 PreProcess 的脚本，在 NLTK 中执行以下管道，将要预处理的文件作为参数 PreProcess(training/267)

1. tokenization (NLTK)
2. sentence splitting (NLTK)
3. POS tagging (NLTK)
4. number normalization
5. date recognition
6. date parsing

1. 标记化（NLTK）
2. 分句（NLTK）
3. 词性标注（NLTK）
4. 数字规范化
5. 日期识别
6. 日期解析

Proofread every step in your pipeline. To save time and gain the anticipated insight, run your script on other texts as well. Solutions that only work on this text and not on others may not get full marks.
校对你的每一步。为了节省时间并获得预期的洞察力，还可以在其他文本上运行脚本。仅适用于本文本而不适用于其他文本的解决方案可能得不到满分。

Description: Each step in your pipeline has specific additional requirements to be considered satisfactory. It is important that you do not limit yourself to the requirements but think beyond the minimum requirements for your solution.
描述：您管道中的每个步骤都有特定的附加要求，可以认为是令人满意的。重要的是，不要将自己局限于需求，而要超越解决方案的最低要求。

> **tokenization** starts with a regular expression-based tokenizer from NLTK. Note that you are free (and possibly required to) improve on that tokenizer for best results. Copious in-line comments in the code for your changes are essential. The enhancements must also be summarized in the report.
> 标记化从 NLTK 中基于正则表达式的标记器开始。请注意，您可以免费（并且可能需要）改进该标记器以获得最佳结果。代码中的大量在线注释对于您的更改至关重要。报告中还必须总结这些改进。

sentence splitting as for tokenization, start with a NLTK module. Enhance if necessary.
句子分割对于标记化，从 NLTK 模块开始。如有必要，加强培训。

POS tagging inspect your options in NLTK. Pick the best module. Do not spend time to improve the POS tagger currently.
词性标记检查 NLTK 中的选项。选择最好的模块。当前不要花时间改进 POS 标记器。

number normalization numbers have their own regular grammars. Number normalization counteracts bad tokenization of numbers. Adapt the tokenizer if necessary and write a number--normalizer grammar to ensure that number segments that the tokenizer split are combined again (i.e., if the tokenizer creates three tokens `1', `.', `2', let the number-normalizer create a single number `1.2') Make sure it does not confuse commas and periods that are part of a number with those that are not. Include any other typographical conventions for numbers. Bonus: create a grammar for numbers written in words (e.g., `three thousand and twelve').
数字规范化数字有自己的规则语法。数字规范化抵消了错误的数字标记化。如有必要，调整标记器并编写数字–规范化程序语法，以确保标记器拆分的数字段再次组合（即，如果标记器创建三个标记“1”、“2”、“1.2”，则让数字规范化程序创建单个数字“1.2”）确保它不会混淆逗号和句点，它们是数字的一部分，而不是数字的一部分。包括数字的任何其他印刷约定。额外好处：为用单词写的数字创建语法（例如“三千零十二”）。

date recognition Create a CFG DateParseCFG, that takes a date string as input, to parse dates. Use POS information and create your own gazetteers. Dates come in different formats, including 2020/9/30, September 3rd, the fifth of November. Do not limit your grammar to these formats. Your grammar should include the nonterminal DATE (as root), DAY, MONTH, YEAR. Aim for large coverage while minimizing the false positives and the false negatives.
日期识别创建一个以日期字符串作为输入的 CFG DateParseCFG 来解析日期。使用 POS 信息并创建自己的地名录。日期有不同的格式，包括 2020 年 9 月 30 日、9 月 3 日和 11 月 5 日。不要将语法限制在这些格式。您的语法应该包括非终结日期（作为根）、天、月、年。以大覆盖率为目标，同时最大限度地减少误报和漏报。

For all modules, create your own test cases for a more general solution.

Deliverables: to be submitted in Moodle by October 3rd, 2020

- (2pts) tokenizer used (enhancements clearly identified in inline comments. This includes the number normalizer.)
  使用标记器（内联注释中明确标识的增强功能。这包括数字规范化器。）
- (2pts) sentence splitter used (any enhancements clearly identified in inline comments)
  使用了分句器（内联注释中明确指出的任何增强）
- (2pts) POS tagger used (any enhancements clearly identified in inline comments)
  使用 POS 标签（内联注释中明确标识的任何增强功能）
- (2pts) *DateParseCFG*
- (2pts) Report: a .pdf document that documents your work and submitted modules
  报告：一个.pdf 文档，用于记录您的工作和提交的模块

- (2pts) Demo File: a .pdf document that shows examples of input and output for all modules and all interesting cases in a single \demo" _le. Organize your demonstration of your demo to be readable. Do not include repetitive example runs. You must select the most helpful runs to show strengths of your modules/grammars. Errors or limitations must be addressed openly in the Report. Note that there are no solutions without errors or limitations.
  演示文件：一个.pdf 文档，在一个\Demo 中显示所有模块和所有感兴趣案例的输入和输出示例“_le。组织演示，使其可读。不要包含重复的示例运行。您必须选择最有用的运行，以显示模块/语法的优势。报告中必须公开说明错误或限制。请注意，没有没有没有错误或限制的解决方案。

Marking scheme: Grading is based on two components. The first component is adherence to the instructions. This is expected to be 100%. The second component is what you bring to the task. We can only appreciate this, if it is well documented in the report and in the code and illustrated with convincing examples in the Demo File. Solutions that work on a wider variety of input data are preferred over narrow solutions (coverage). Solutions that produce fewer false positives are preferred (specificity). There is no optimum and you must explain your choices.
评分方案：评分基于两个部分。第一个要素是遵守说明。这一比例预计为 100%。第二个部分是您为任务带来的内容。只有在报告和代码中有很好的文档记录，并在演示文件中用令人信服的示例加以说明，我们才能理解这一点。与狭义的解决方案（覆盖范围）相比，更倾向于使用范围更广的输入数据的解决方案。首选产生较少误报的解决方案（特异性）。没有最佳选择，你必须解释你的选择。