Computational Linguistics
Fall 2018

Passive Voice Detection - Submission Guideline

(DDL: Dec. 2th)

In this task, you are going to design an algorithm in order to determine if a given sentence contains the usage of the passive voice.
Discussion or search on the Internet is allowed, but please write the code yourselves, and **cite all resources** that you've looked at in the report. Copying code will result in the assignment being discarded at the very least.

Dataset description

There is no training set for this assignment, but only a test set, which is hidden. On Nov. 26th, we'll release the first part (~20%) of the test set as a **trial set**, and you can try submitting your code to see how it performs on it. After the deadline, we'll test your code on the full **test set**, and grade your algorithm according to the final performance.
The difficulty of the final test set might be higher than the trial set, depending on how the class performed on the trial set.

Sample test cases

A test case is a sentence with its ground-truth tag, 'y' if it contains the passive voice, 'n' if not.

- **Sentence**: This vulnerability is known as Cloudbleed.
  Tag: y
- **Sentence**: The thrift holding company said it expects to obtain regulatory approval and complete the transaction by year-end.
  Tag: n
- **Sentence**: Alan Spoon, recently named Newsweek president, said Newsweek's ad rates would increase 5% in January.
  Tag: y
  (Note that we do not only detect the usage of the passive voice in the main clause of the sentence. A past participle used as a postpositive attribute / 后置定语 is also considered as a usage of the passive voice, which is called a reduced passive voice construction, i.e., a passive verb without passive auxiliary. There will be a **very small** proportion of such sentences in the test set.)

What you should include in your code

Your code should have a function named **predict()** as follows:

```
def predict(sentence):

    '''Given a sentence, determine if it contains the passive voice
    :param sentence: a string, the sentence to be classified
    :return: a string, ’y’ if the sentence contains the passive voice, else ’n’
    '''

    #########################################
    #        Complete this function         #
    #########################################

    return …
```

Note:

- External libraries you can use include **nltk** and **Python’s standard library** (see a list here). If you’d like to import additional libraries, please first ask the instructors for permission.

Deliverables

- Your code: it should be a file named **main.py**.
- Your report: it should be in **PDF** format, including but not limited to **methodology, algorithm, difficulties and possible solutions, error analysis**, **references (data and resources used),** etc.

Note:

- Please **drag and drop** all files in the submission box for the Passive Voice Detection Assignment on GradeScope, and **do not zip** anything.
- If running the **predict()** function depends on any external files (e.g. a file of word list, named ‘wordlist.txt’), please submit the files along with your code and report (still, do not zip anything). Ensure that you use **relative path** (e.g. ‘./wordlist.txt’) instead of absolute path (e.g. ‘/Users/YourName/Documents/CL/passive/wordlist.txt’) when opening the files in your code.
- You can submit as a team (choose a cool leaderboard name!). Only one team member needs to submit, and he/she can add other members by clicking ‘Add Group Member’ on the Autograder Results page on GradeScope.
- You can submit multiple times before the deadline. Each time you submit your code, the autograder will call the **predict()** funciton in **main.py**, and you can see the performance (F1 score) of your algorithm on the **LeaderBoard**, as well as your position among all teams.

What you’ll be graded on

- 40%: Performance on the final leaderboard
- 5%: Code Style (clarity, readability, and elegance)
- 45%: Report
- 10%: Presentation