Introduction to Software Engineering Project

Authorship Detection

Code and Report due: Midnight Friday May 31st (end Week 15)

Report

Written in Overleaf, saved as a pdf (`Group_number_groupname.pdf`) and uploaded to Moodle by Midnight Friday May 31st.

Write a reflection on working on this project as a team.

– how did you approach the problem

– describe your team work, team roles, etc

– how did you make your group work

– what did you learn

What else did you learn, for example, how about other CS ideas like weighted averages, natural language understanding, machine learning and the challenges of working with real data rather than clean and tidy toy input?

Can you think of ways to improve this project in the future for example, including new linguistic features such as frequency of various punctuation marks or some common words?

How many of the mystery files does your program guess correctly?

If your program got some wrong, why do you think this might be?

Submitting your report

For this assignment, submit two files to Moodle:

- `find_author.py`
- `Group_number_groupname.pdf`

Once you have submitted, be sure to check that you have submitted the correct version. Remember that spelling of filenames, including case, counts. **Your file must be named exactly as above.**

Future Thoughts

If you carefully examine the mystery files and correctly code up your assignment, you'll see that it correctly classifies many of the documents – but not all. Some of the linguistic features that we have used (type-token ratio and hapax legoamana ratio in particular) are standard techniques, but they may not be sufficient to do the classification task. There are other standard features and more sophisticated techniques that were too complicated for this assignment. Suppose you could use a Python dictionary to keep a count of how many times each word appeared in a document. What new linguistic features would that allow you to use? Some authors like to use lots of exclamation marks!!! or perhaps just a lot of punctuation?! Could you devise a linguistic feature to measure this? While this is fun to think about, do **not** add any different linguistic features to the version of the program that you submit for marking.

Another area for thought is how authorship attribution is related to plagiarism detection. In this TIME magazine article, plagiarism detection software is used to support the claim that Shakespeare was the author of an unattributed play.

In the field of machine learning (of which authorship detection is a subfield), programs often learn their own configuration values through training. In our case, could the program learn from trying to guess an author and then being told the right answer? Could it learn to adjust the weights being applied to the different features? What about learning a new feature? How would you do this?

Connection to Current CS Research and Development

Automated authorship detection has uses in plagiarism detection, email-filtering, social-science research and as forensic evidence in court cases. Also called authorship attribution, this is a current research field and the state-of-the-art linguistic features are considerably more complicated than the ones calculated in this assignment.