WASP Software Engineering - Assignment 1

Maarten Jensen - Umeå University

June 23, 2022

I did a bachelor in Computer Engineering where we used for example a code style guide[1], but also making various UML diagrams and making Unit tests for assuring code quality. However I never used a static analysis tool before, maybe because back then (7 years ago) it was not so popular. Or because in general software engineering (not machine learning algorithms) it is maybe more popular to use other tools to assure code quality.

This was my first using a static analysis tool. However the pynblint tool was easy to install and easy to use as it just entails running a single command. We applied it to two of my notebook projects which were both related to machine learning as they are from a WASP machine learning course. The first project didn't gave any remarks. The second project actually did give some remarks. I learned that pynblint can actually give remarks about the structure and can help making the notebook better and therefore less prone to errors. Me and Luigi talked about how some remarks were probably false positives and that it depends a lot on the project whether the remark of pynblint should actually be fixed. And also pynblint seems to be most relevant for a team working and sharing notebooks (to assure some quality).

Personally the pynblint tool will not be useful for my PhD project since I do not work with notebooks. In my project I make simulations of human behavior which especially with more complex simulations (which I'm focusing on) does not work that well in notebooks since there is much complexity in the code and an IDE like pycharm is much more suitable. Although perhaps for demonstrations of a simulation a python notebook could be used. As then it would be easy to adjust the parameters of the model through the notebook and plot interesting graphs, while the simulation model itself is imported into the notebook.

[1] https://www.stroustrup.com/JSF-AV-rules.pdf

I thought pynblint was already very intuitive and easy to use. One think I thought about was integrating pynblint into jupyter notebooks to make the overhead even smaller. As I think the overhead of using something like this may hinder people (especially researchers) from using the tool regularly.

I did not see the use of pynblint specifically for machine learning. Since pynblint did not give specific remarks on the content of the code rather than the style of the code. While it seems to be focused on machine learning prototypes [2] I would argue it can be applied to python notebooks in general. Pynblint seems to not dive into the code and provides feedback on the notebook structure rather than the 'machine learning' code. It could thus be improved by trying to understand the code, libraries used or data used however I think developing this provides a whole new challenge.

[2] https://arxiv.org/abs/2205.11934