Why shouldn't I just use Python code for configuration?

It is relatively common for many applications to avoid the use of a different markup language for configuration and simply allow configuration to be done using code. One famous and unapologetic example of this is Django, which requires all configuration to be in a "settings.py" or similar file.

This seems like a great idea from the outset - Python is more flexible than any configuration language, so, for instance, if you wanted to use a list comprehension or read a file or call an API to fill a value, you can.

However, with this flexibility comes many traps and unsightly pitfalls. The Django pitfalls in particular are cogently summed up by Ned Bachelder on his blog - pitfalls which have been the cause of countless bugs over the years.

The language expressiveness trade off applies at every level in code

- We need less powerful languages.
- Rule of least power (wikipedia).
- Principle of least power by Tim Berners Lee.
- Principle of least power by Jeff Atwood (coding horror blogger / stack overflow founder).

A good way of refactoring, in fact, is to take a large chunk of Turing-complete Python code that *can* be transformed directly into StrictYAML with no loss in expressiveness and and to transform it - for example, a list of translation strings, countries or parameters.

It also makes it easier to have the markup generated by another program or a templating language. While you technically *can* do this with Turing-complete code, it will often lead to a debugging nightmare - just ask C++ programmers!