Step 2: Create a configuration file

Now that we have our prototype in a notebook, we can create a single configuration file. Up to this point we have different configuration components spread throughout the notebook. You might have the location of your datasets in one cell of code, the modules you are importing in another, and the path you're saving your model to maybe even directly in your code.

We would prefer to have all of this in one file so that we know where to go when we want to change a configuration component. This makes running experiments much easier: rather than having to change code within the notebook, an experiment could be a change to just one line in the configuration file.

A common way to organise your configuration is by creating a `params.yaml` file. We remove any hardcoded parameters and paths from the code and store them in this file. We then replace the hardcoded parameters in our notebook with references to the parameters in the dedicated file.

> Note: For the purposes of this course we will stick to a single configuration file. When the complexity of your project increases, you could split up `params.yaml` into multiple files. You can use as many configuration files as you want with arbitrary names, as long as they are in a supported format.

We recommend creating some logical groups of parameters in the configuration file, for example by pipeline stage. First list the parameters needed for data loading, for example, and then follow up with groups for training and testing. Oftentimes it's also convenient to have a `base` config group with parameters shared across several stages, such as seeds.

Here's an example of a simple `params.yaml` file:

```yaml
base:
    seed: 42

data_load:
    dataset_csv: 'data/raw/iris.csv'

data_split:
    test_size: 0.2
    trainset_path: 'data/processed/train_iris.csv'
    testset_path: 'data/processed/test_iris.csv'

train:
    cv: 3
    estimator_name: logreg
    C: 0.001
    max_iter: 100
    solver: 'lbfgs'
    multi_class: 'multinomial'
    model_path: models/model.joblib
```

After creating this file we can access the parameters from our notebook like so:

```python
import yaml

with open('params.yaml') as conf_file:
    config = yaml.safe_load(conf_file)

print(config['data_load']['dataset_csv'])
# >> 'data/raw/iris.csv'
```