Common pitfalls and recommended practices

This chapter will explore the common struggles with scikit-learn: managing data preprocessing, sneaking, and uncertainty. A significant issue of this kind is the lack of uniform preprocessing — if you make changes on train data (like scaling), it should be performed identically with test data, and otherwise you will get a worse result from your model. Pipelines also enable automation, and reduce data leakage — the condition in which training of a model is impacted by test data, causing results to be overly optimistic. Feature selection is to be done based only on the training data, not over the test data. The other difficulty for any of this has to do with managing randomness. Using an Integer to specify ‘random_state’ (thus replicable result) although at the potential cost of some cross-validation results coef_ instability. Using instances of RandomState adds some variation and makes the model more robust to randomness by testing it in different random situations. Some Estimators, like RandomForest or the cross-validation splitter, behave differently depending on their “random_state”, which makes results across folds not exactly because of the variability contributed by them. Avoid using 'random_state=None' and use best practices on dealing with randomness in the CV processes, and distribution of RandomForestClassifier or any random estimator.