Hypertunning the Random Forest

Adjusting the setting of an algorithm to optimize the performance of an algorithm. The parameters of a random forest are the variables and thresholds used to split each node learned during training.

Using the default setting provided in the scikit-learn library is not optimal in all cases so, we need to perform experimentation with the given classification algorithm to fine tune the Model. So we try the different setting available for the certain model.

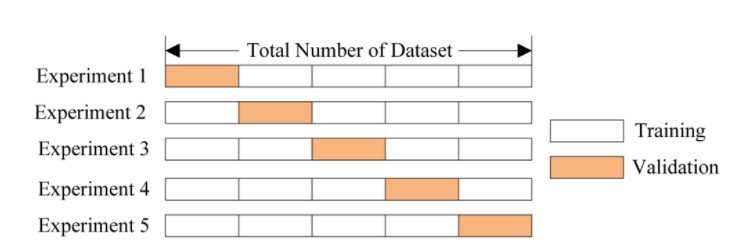
Evaluation of the model on the training sets we have the common problem also known as the overfitting.

Overfitting: - When the model performs best in the training data sets and worse on the testing datasets then such models are said to be overfitted.

A overfitted model looks great on the training sets but when the real application comes into play it performs poor. So, we need to perform cross validation.

Cross Validation:

When we approach a machine learning problem, we make sure to split our data into a training and a testing set. In K-Fold CV, we further split our training set into K number of subsets, called folds. We then iteratively fit the model K times, each time training the data on K-1 of the folds and evaluating on the Kth fold (called the validation data). As an example, consider fitting a model with K = 5. The first iteration we train on the first four folds and evaluate on the fifth. The second time we train on the first, second, third, and fifth fold and evaluate on the fourth. We repeat this procedure 3 more times, each time evaluating on a different fold. At the very end of training, we average the performance on each of the folds to come up with final validation metrics for the model.



Random Search Cross Validation in Scikit-Learn

Using Scikit-Learn’s RandomizedSearchCV method, we can define a grid of hyperparameter ranges, and randomly sample from the grid, performing K-Fold CV with each combination of values.