Question 1

Which hyperparameters are first to tune in sklearn's RandomForest?

Correct answers:

- n_estimators, max_depth, min_samples_split Yes! These parameters are important. The first one should just be sufficiently large, you do not actually need to tune it.

Incorrect answers:

- n_jobs, random_state, verbose. Some of these parameters can even change the result of the training but only because of randomness involved. They are for sure not the parameters to tune.
- bootstrap, oob_score, warm_start. These parameters are not what you want to tune in the model!

Question 2

Suppose you fit LightGBM to your train data and check performance on the validation set. The train set consists of 500 rows and 1000 different features and validation set consist of 50 objects. You run automatic hyperparameter optimization method overnight and in the morning you select the best parameters, produce results for the test set and submit to the leaderboard. We also know that test set comes from the same distribution as train and validation sets.

Correct answers:

- There is a high chance of overfitting to the validation set. That is, there is a high chance that score on the test set will be bad. This is because we've tried too much hyperparameters while the dataset is small and the number of features is large. *Correct!* This is because of multiple comparisons fallacy.

Incorrect answers:

- There is a low chance of overfitting to the validation set. That is, there is a high chance that score on the test set will be good. This is because we found good parameters on validation set and test set is similar to the test set. The fact that test and validation sets come from the same distribution does not make them similar in terms of optimal hyperparameters.

Question 3

Suppose you want to find a good set of hyperparameters for a dataset with 1000 points and have resources to do fitting 2000 times. Which method of model selection your should use?

Correct answers:

- k-Fold scheme (i.e. split data into k part, use k-1 parts for training and the last one for quality estimation; repeat for each part). Yes! It is easy to note than we will check 2000 / k sets of hyperparameters -- this number still big enough (for typical k in [3,..,10]), but this scheme is much more robust.

Incorrect answers:

- Hold-Out scheme (i.e. divide data into two parts, use first for model fitting and second for estimation of quality). See previous question for detailed explanation.
- Select model by quality on training set (i.e. fit model on whole dataset and measure quality on the same data). This is overfitting. Such way cannot be used to estimating quality of model.
- Leave-one-out validation (i.e. fit model for all points except one and estimate quality for this single point; repeat for every point). Since LOO requires 1000 fittings for single set of hyperparameters, we can estimate only 2 combinations of hyperparameters -- it's way too small.

Question 4

Suppose you train Neural Network with SGD and see that it overfits data. Which of the following actions can help you to regularize model?

Correct answers:

- Insert (or increase rate of) Dropout layers inside NN. Yes, Dropout is a known way to regularize model via dropping some randomly selected features at every step.
- Add (or increase) Weight Decay. Weight Decay is essentially the same thing as L2 Regularization.
- Reduce number of parameters (e.g. remove some layers) Reducing number of parameters make model less flexible and harder to overfit

Incorrect answers:

- Change optimization method to Adam. Adam is a better optimizer than SGD, thus overfits training data better and faster.
- Add more layers. No, this will increase complexity of neural net and do not introduce any regularization effect.

Mark as completed