|  |  |
| --- | --- |
| 1 | Is there any way to combine five different models that have all been trained on the same training data and have all achieved 95 percent precision? If so, how can you go about doing it? If not, what is the reason? |
| Ans. | you can try combining them into a voting ensemble, which will often give you even better results. It works better if the models are very different (e.g., an SVM classifier, a Decision Tree classifier, a Logistic Regression classifier, and so on). It is even better if they are trained on different training instances (that’s the whole point of bagging and pasting ensembles), but if not it will still work as long as the models are very different. |
| 2 | What's the difference between hard voting classifiers and soft voting classifiers? |
| Ans. | A hard voting classifier just counts the votes of each classifier in the ensemble and picks the class that gets the most votes. A soft voting classifier computes the average estimated class probability for each class and picks the class with the highest probability. This gives high-confidence votes more weight and often performs better, but it works only if every classifier is able to estimate class probabilities (e.g., for the SVM classifiers in Scikit-Learn you must set probability=True). |
| 3 | Is it possible to distribute a bagging ensemble's training through several servers to speed up the process? Pasting ensembles, boosting ensembles, Random Forests, and stacking ensembles are all options. |
| Ans. | * Bagging - Yes it is possible to train distributedly in several serves since each predictor in the ensembles is independent of the others. * The same goes with pasting ensembles and random forests * Boosting - each predictor in a boosting ensemble is built based on the previous predictor, so training is sequential. It is not possible to train them in different serves. * Stacking - All the predictors in the given layers are independent of each other, so they can be trained in parallel on multiple serves. But training has to be done only once the previous layer is done. |
| 4 | What is the advantage of evaluating out of the bag? |
| Ans. | out of the bag, instances are those who are not picked for training. These are used for evaluating the model without the need for an additional validation set. With this, it is possible to have an unbiased evaluation of the ensembles. With Out of bag instances, we will have more instances available for training and our ensemble can perform better. |
| 5 | What distinguishes Extra-Trees from ordinary Random Forests? What good would this extra randomness do? Is it true that Extra-Tree Random Forests are slower or faster than normal Random Forests? |
| Ans. | When you are growing a tree in a Random Forest, only a random subset of the features is considered for splitting at each node. This is true as well for Extra- Trees, but they go one step further: rather than searching for the best possible thresholds, as regular Decision Trees do, they use random thresholds for each feature. This extra randomness acts like a form of regularization: if a Random Forest overfits the training data, Extra-Trees might perform better. Moreover, since Extra-Trees don’t search for the best possible thresholds, they are much faster to train than Random Forests. However, they are neither faster nor slower than Random Forests when making predictions. |
| 6 | Which hyperparameters and how do you tweak if your AdaBoost ensemble underfits the training data? |
| Ans. | To fix the AdaBoost ensemble underfits the training data,   * increasing the number of estimators * reducing the regularization hyperparameters of the base estimator * try slightly increasing the learning rate |
| 7 | Should you raise or decrease the learning rate if your Gradient Boosting ensemble overfits the training set? |
| Ans. | To fix if Gradient Boosting ensemble overfits the training set,   * decreasing the learning rate. * use early stopping to find the right number of predictors |