

sepal width (cm) 0.0231192882825
petal length (cm) 0.441030464364
petal width (cm) 0.423357996355

Similarly, if you train a Random Forest classifier on the MNIST dataset (introduced in [Chapter 3](#)) and plot each pixel's importance, you get the image represented in [Figure 7-6](#).

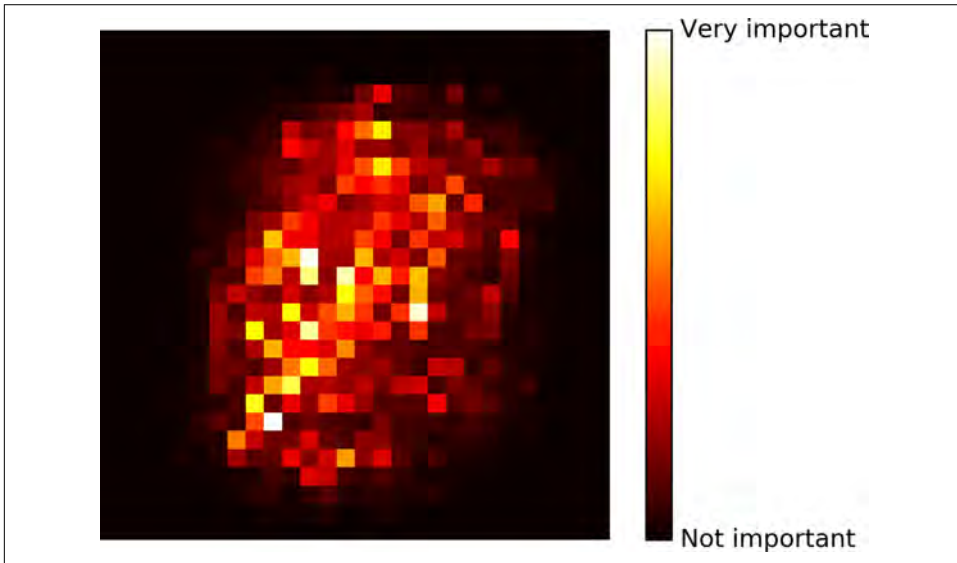


Figure 7-6. MNIST pixel importance (according to a Random Forest classifier)

Random Forests are very handy to get a quick understanding of what features actually matter, in particular if you need to perform feature selection.

Boosting

Boosting (originally called *hypothesis boosting*) refers to any Ensemble method that can combine several weak learners into a strong learner. The general idea of most boosting methods is to train predictors sequentially, each trying to correct its predecessor. There are many boosting methods available, but by far the most popular are *AdaBoost*¹³ (short for *Adaptive Boosting*) and *Gradient Boosting*. Let's start with AdaBoost.

¹³ "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," Yoav Freund, Robert E. Schapire (1997).

AdaBoost

One way for a new predictor to correct its predecessor is to pay a bit more attention to the training instances that the predecessor underfitted. This results in new predictors focusing more and more on the hard cases. This is the technique used by AdaBoost.

For example, to build an AdaBoost classifier, a first base classifier (such as a Decision Tree) is trained and used to make predictions on the training set. The relative weight of misclassified training instances is then increased. A second classifier is trained using the updated weights and again it makes predictions on the training set, weights are updated, and so on (see [Figure 7-7](#)).

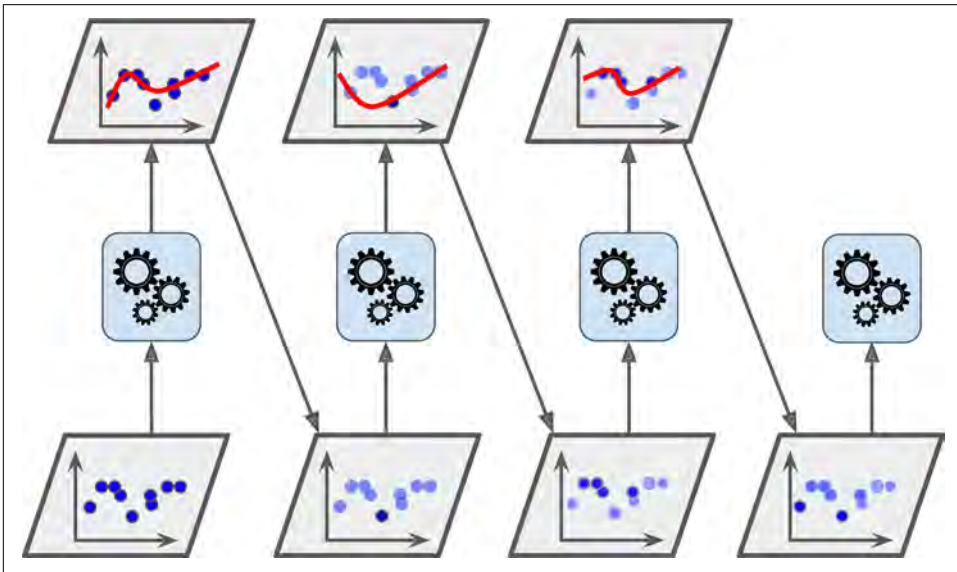


Figure 7-7. AdaBoost sequential training with instance weight updates

[Figure 7-8](#) shows the decision boundaries of five consecutive predictors on the moons dataset (in this example, each predictor is a highly regularized SVM classifier with an RBF kernel¹⁴). The first classifier gets many instances wrong, so their weights get boosted. The second classifier therefore does a better job on these instances, and so on. The plot on the right represents the same sequence of predictors except that the learning rate is halved (i.e., the misclassified instance weights are boosted half as much at every iteration). As you can see, this sequential learning technique has some similarities with Gradient Descent, except that instead of tweaking a single predictor's

¹⁴ This is just for illustrative purposes. SVMs are generally not good base predictors for AdaBoost, because they are slow and tend to be unstable with AdaBoost.