DSC 255: Machine learning

Week 9 — Solutions

1. *Other methods with the expressive power of decision trees.*

   (a) Linear classifiers cannot represent any boundary that is not linear.

   (b) Support vector machines with a quadratic kernel cannot represent any boundary that is not quadratic.

   (c) Nearest neighbor can capture any boundary.

   (d) Classifiers based on Gaussian generative models end up with quadratic boundaries and are thus not as expressive as decision trees.

2. There are d features, and $n-1$ split points for each feature (look at the one-dimensional values of that feature alone, in sorted order; the split will lie between two consecutive values). Thus the total number of possible splits is $d(n-1)$.

3. When $p = 0.2$, the Gini index is $2 \times 0.2 \times 0.8 = 0.32$.

4. *Working with weighted data.* Suppose we have data points $(x_1, y_1), \ldots, (x_n, y_n)$, with weights $\lambda_1, \ldots, \lambda_n > 0$. Suppose the possible labels are $\{1, 2, \ldots, k\}$.

   (a) At any node of the data, let $S \subset [n]$ denote the subset of points reaching that node. We compute the proportions of the different labels $(p_1, \ldots, p_k)$ using the weights of these points:

   $$p_j = \frac{\sum_{i \in S} \lambda_i \cdot 1(y_i = j)}{\sum_{i \in S} \lambda_i}.$$

   The impurity of a split (e.g., using the Gini index) is then a function of these p_j values.

   (b) To compute the mean and covariance for class j, use the weights:

   $$\mu_j = \frac{\sum_{i=1}^n \lambda_i \cdot 1(y_i = j) x_i}{\sum_{i=1}^n \lambda_i \cdot 1(y_i = j)}, \ \Sigma_j = \frac{\sum_{i=1}^n \lambda_i \cdot 1(y_i = j)(x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=1}^n \lambda_i \cdot 1(y_i = j)}.$$

   (c) Include weights in the SVM optimization: e.g., for the binary case, use

   $$\begin{aligned} \min_{w \in \mathbb{R}^d, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \quad & \|w\|^2 + C \sum_{i=1}^n \lambda_i \xi_i \\ \text{s.t.:} \quad & y_i(w \cdot x_i + b) \geq 1 - \xi_i \quad \text{for all } i = 1, 2, \ldots, n \\ & \xi \geq 0 \end{aligned}$$

5. *Convergence behavior of boosting.*

   (a) There is no guarantee that boosting will converge to a model with zero test error. Such a model need not even exist, e.g., in situations with inherent uncertainty (recall our discussion of cases where perfect classification might not be possible).

(b) True, boosting will converge to a model with zero training error.

(c) False, because boosting does not converge to a model in H. Its model is a linear combination of classifiers from H.

6. *Random forests versus boosted decision trees.*

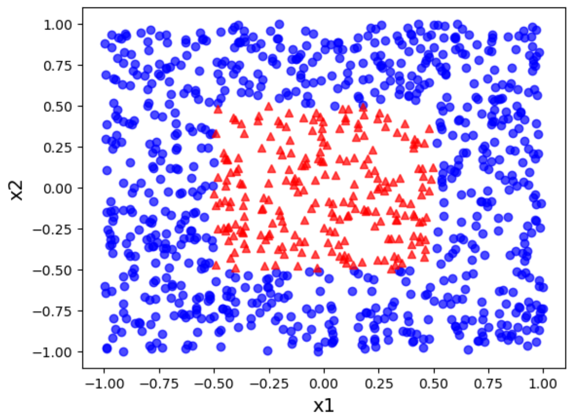
(a) True, random forests can be trained in parallel whereas in boosted decision trees, the trees must be trained sequentially.

(b) False: each individual tree in a random forest is not more highly optimized.

(c) False: each individual tree in a random forest need not have better accuracy.

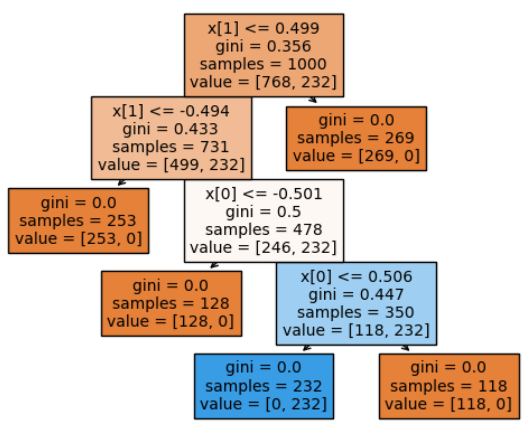
7. *A toy 2-d data set for decision trees and boosting.*

(a) Here is the data from `mini-data.txt`.

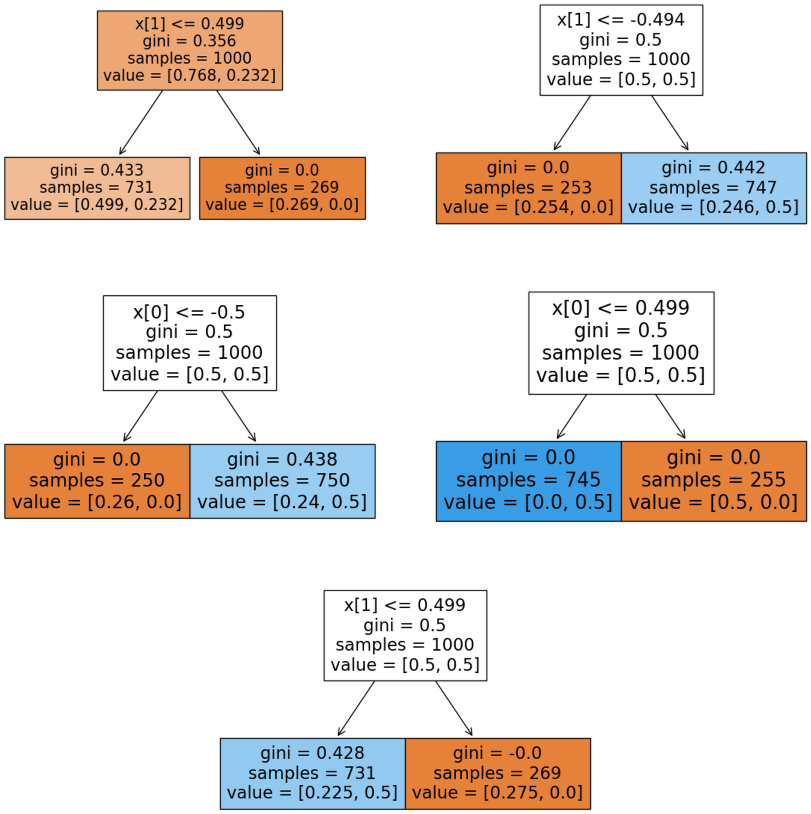


(b) There are many reasonable stopping criteria. We used `max_depth = 4`.

(c) Here is the decision tree.



(d) We set the boosting iterations to 5. Here are the stumps.



(e) Here is a table of training accuracy with each successive stump.

# Stumps	Accuracy
1	0.768
2	0.768
3	0.882
4	1.000
5	1.000

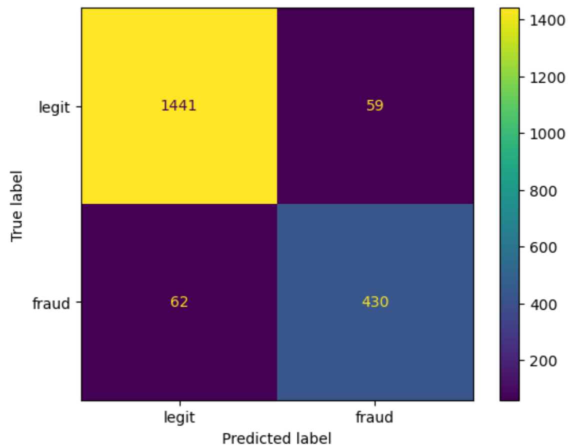
8. *Credit card fraud data.*

   (a) Out of the 284,807, only 492 are fraudulent. This is problematic because a classifier can get low error by always predicting legitimate.

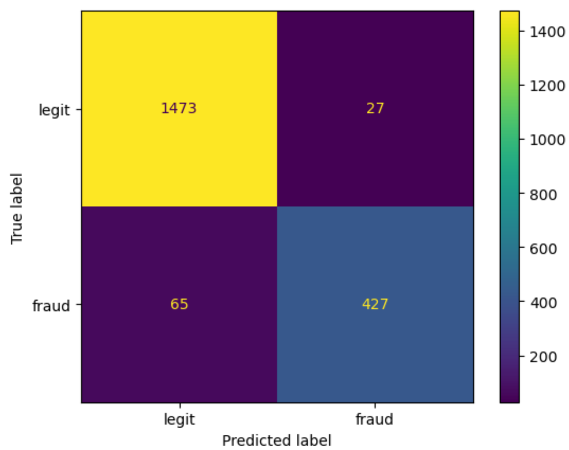
   (b) We retain all 492 fraudulent transactions and subsample just 1500 legitimate transactions. This gives a training set of size 1992.

   (c) Here are the confusion matrices for the three methods.

   Decision tree:



Boosted decision stumps:



Random forest: