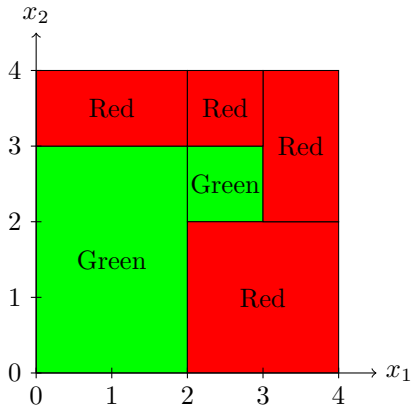
Tree-Based Methods

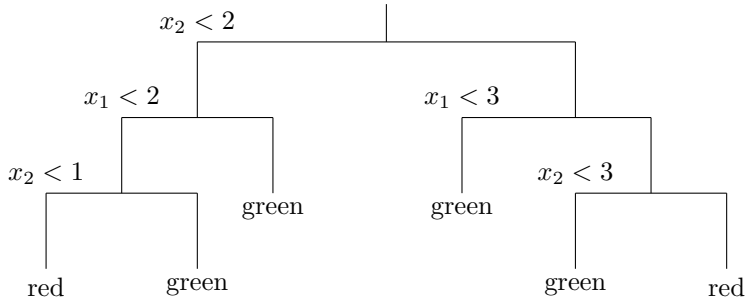
7.1 Basics of decision tree

(a) Sketch (by hand) the classification tree corresponding to the following partition:



(b) How many terminal nodes (leaves) and internal nodes does the resulting tree have?

(c) Sketch (by hand) the partition of $[0, 4] \times [0, 4]$ for the following classification tree



7.2 Bootstrapping for bagging

Bagging starts with boostrapping B datasets $\{\widetilde{\mathcal{T}}^b\}_{b=1}^B$ from the original training dataset $\mathcal{T}$ (i.e., independently drawing data points with replacement from $\mathcal{T}$). Let us say that all data the original dataset $\mathcal{T}$ as well as the boostrapped ones $\mathcal{T}^b$ contains n data points.

(a) What is the probability that the first data point in the first bootstrapped dataset $\widetilde{\mathcal{T}}^1$ is *not* the ith data point from the original dataset $\mathcal{T}$?

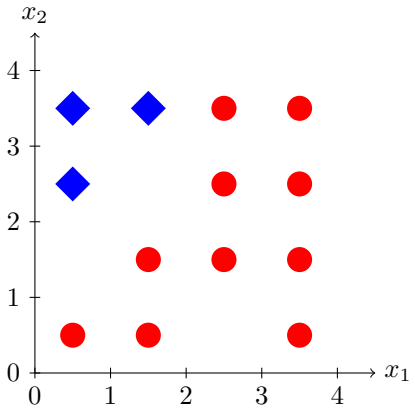
(b) What is the probability that the first dataset $\widetilde{\mathcal{T}}^1$ does *not* contain the ith data point from the original dataset $\mathcal{T}$ at all?

(c) In the limit as $n \to \infty$, how many of the data points in $\mathcal{T}$ can you expect to find replicated at least once in $\widetilde{\mathcal{T}}^1$?
Hint: $(1 + x/n)^n \to e^x$ *as* $n \to \infty$

(d) As we have seen above a certain data point in $\mathcal{T}$ is not present in all bootstrapped datasets $\widetilde{\mathcal{T}}^1, \ldots, \widetilde{\mathcal{T}}^B$. How can this be used instead of cross-validation in bagging methods?

7.3 Tree construction: classification error rate vs. Gini index

Consider the following data (different colors and shape represent different classes), for which we are going to construct a classification tree:



There are several alternatives in how to decide which splits to make when constructing a classification tree. Trees are often constructed by adding one split at a time to the tree, and terminate the construction when some criterion is reached. When deciding which split to add to the tree, a cost function

$$\sum_{\ell=1}^{|T|} n_\ell Q_\ell \tag{7.1}$$

is considered, where T is the tree object, $|T|$ is the total number of terminal nodes in the tree, n_ℓ is the number of training data points in the region corresponding to the terminal node ℓ, and Q_ℓ is a node impurity measure of the same region. Two common measures are

$$\text{Classification error rate: } Q_\ell = 1 - \max_m \hat{\pi}_{\ell m}, \tag{7.2}$$

$$\text{Gini index: } Q_\ell = \sum_{m=1}^{M} \hat{\pi}_{\ell m}(1 - \hat{\pi}_{\ell m}), \tag{7.3}$$

where $\hat{\pi}_{\ell m}$ is the proportion of training data points with label m in the region corresponding to node ℓ. Since the classification for any data point within a certain region is decided with a 'majority vote', the proportion of correct training data classifications in region ℓ can thus be expressed as $\max_m \hat{\pi}_{\ell m}$, and the proportion of misclassifications as $1 - \max_\ell \hat{\pi}_{\ell m}$.

(a) Consider the problem of making the first split when building a tree. Where would you make the split?

  (i) Find the single split which minimizes the cost function (7.1) when considering the classification error rate.

  (ii) Find the single split which minimizes the cost function (7.1) when considering the Gini index.

  Which split would be the optimal (counting the number of misclassifications) if you would terminate the tree-construction now, and be satisfied with this depth-1 tree (i.e., stump)?

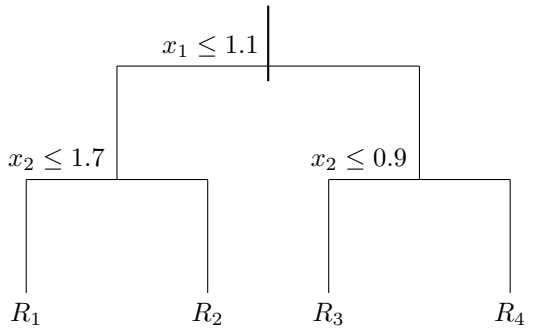
(b) Now add another split to your trees (i) and (ii), respectively. Which tree seems to be the preferable? Explain your findings!

7.4 **Regression trees**

Consider a regression problem with a two input variables $\mathbf{x} = [x_1\ x_2]^\mathsf{T}$, where $x_1 \in [0, 3]$ and $x_2 \in [0, 3]$, and one output $y \in \mathbb{R}$. Based on the following training data

i	x_1	x_2	y
1	1.4	1.4	0.1
2	2.2	2.2	0.6
3	0.2	0.8	-0.7
4	1.0	2.8	-1.8
5	0.6	0.2	0.3
6	0.4	2.2	-1.9
7	0.6	1.4	-0.9
8	1.2	1.8	-0.5
9	1.2	0.6	0.7
10	1.8	0.6	1.5

the regression tree shown below has been constructed using recursive binary splitting.



(a) Draw the corresponding input partitioning to this tree. Mark the regions with the names of the leaf nodes $R_1, \ldots, R_4$.

(b) Use the regression tree to predict the output of the test input $\mathbf{x}_\star = [1.5\ 1.8]^\mathsf{T}$

(c) Continue to grow the tree by adding splits such that there are at most two data points in each region. Choose your splits in order to minimize the residual sum of squares (RSS)

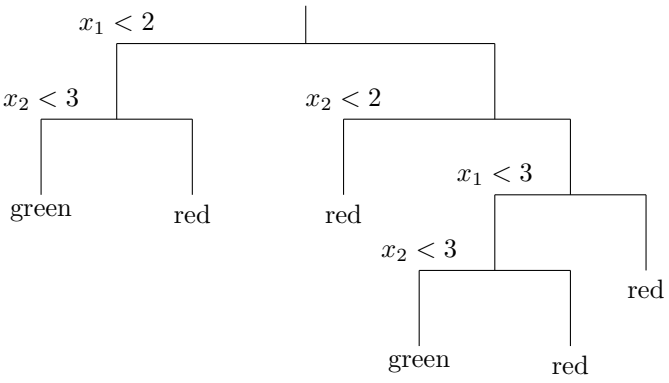
$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \widehat{y}_{R_j})^2, \tag{7.4}$$

where J is the number of regions/terminal nodes and $\widehat{y}_{R_j}$ is the mean value of all training outputs in that region (which also is the prediction for test points in that region). Which splits do you make?

(d) What is the predicted output of the test input $x_\star$ from (b) using this new, deeper tree?

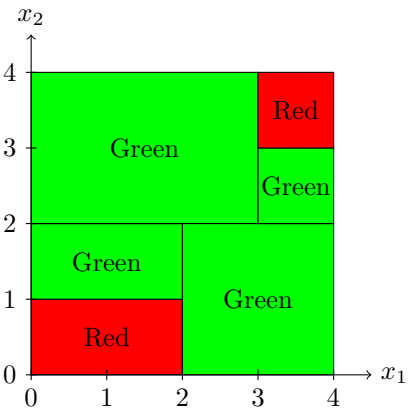
Solutions

7.1 (a)



(b) The tree has six terminal nodes and five internal nodes.

(c)



7.2 (a) The probability that the first data point in $\widetilde{\mathcal{T}}^1$ is the ith data point from $\mathcal{T}$ is $1/n$, hence is its complement $1-1/n$.

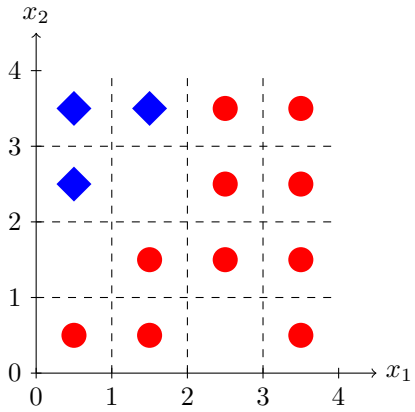
(b) Since the data points in $\widetilde{\mathcal{T}}^1$ are drawn independently from $\mathcal{T}$, the probability for the ith data point never to be drawn in n draws is $(1-1/n)^n$.

(c) Since $(1-1/n)^n \to e^{-1} = 1/e$ as $n \to \infty$, the probability for the ith data point from $\mathcal{T}$ not to be replicated in $\widetilde{\mathcal{T}}^1$ is $1/e$, and we hence expect that $1-1/e \approx 63\%$ of the data points from $\mathcal{T}$ to be replicated in $\widetilde{\mathcal{T}}^1$ on the average.

(d) Remember, the main idea in validation is to see how well a method performs on previously unseen data points (validation data or test data). In cross validation, this is achieved by holding out a fraction of the training data, and use it as validation data instead. In bagging, a similar effect can be achieved without the extra computational burden of cross validation (useful in situations when the computational power is scarce).

In bagging, a separate model is learned for each bagged dataset $\widetilde{\mathcal{T}}^b$, and the final prediction is made as an average over the predictions from *all* B models. If we want to estimate the quality of the prediction for the ith data point in the original set $\mathcal{T}$ without using cross-validation, we can make a prediction by averaging only over the approximately $(1-0.63)B = 0.37B$ models estimated without the ith data point, and compare it to the true y_i.

7.3 (a) There are infinitely many possible splits we could make, but if we restrict ourselves to splits at integer values in the region where the training data is located, we have 6 sensible options (either at $x_1 = 1, 2$ or 3 or $x_2 = 1, 2$ or 3).



To avoid unnecessary calculations, we can start by ruling out $x_1 = 3$ and $x_2 = 1$, since they are clearly suboptimal.

(i) When considering misclassification rate $Q_\ell(T) = 1 - \max_m \widehat{\pi}_{\ell m}$, the cost function take the following values for the different splits:

Split	n_1	$\widehat{\pi}_{1B}$	$\widehat{\pi}_{1R}$	$Q_1(T)$	n_2	$\widehat{\pi}_{2B}$	$\widehat{\pi}_{2R}$	$Q_2(T)$	$n_1Q_1(T) + n_2Q_2(T)$
$x_1 < 1$	3	2/3	1/3	1/3	10	1/10	9/10	1/10	2
$x_1 < 2$	6	3/6	3/6	3/6	7	0/7	7/7	0	3
$x_2 < 2$	6	0/6	6/6	0	7	3/7	4/7	3/7	3
$x_2 < 3$	9	1/9	8/9	1/9	4	2/4	2/4	2/4	3

Thus, we take the decision to make the split at $x_1 < 1$.

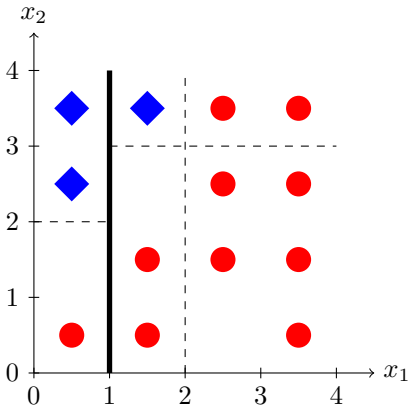
(ii) When considering the Gini index $Q_\ell(T) = \sum_{m=1}^{M} \widehat{\pi}_{\ell m}(1 - \widehat{\pi}_{\ell m})$, the cost function take the following values for the different splits:

Split	n_1	$\widehat{\pi}_{1B}$	$\widehat{\pi}_{1R}$	$Q_1(T)$	n_2	$\widehat{\pi}_{2B}$	$\widehat{\pi}_{2R}$	$Q_2(T)$	$n_1Q_1(T) + n_2Q_2(T)$
$x_1 < 1$	3	2/3	1/3	4/9	10	1/10	9/10	9/50	3.13
$x_1 < 2$	6	3/6	3/6	1/2	7	0/7	7/7	0	3
$x_2 < 2$	6	0/6	6/6	0	7	3/7	4/7	24/49	3.43
$x_2 < 3$	9	1/9	8/9	16/81	4	2/4	2/4	1/2	3.78

Thus, we take the decision to make the split at $x_1 = 2$.

If this is the only decision we are going to make, the decision based on the classification error rate is also the optimal decision for minimizing the classification error rate (only two training data points are misclassified).

(b) (i) For the misclassification-index based tree, the relevant splits to make are the ones marked with dashed lines (again, other splits are also possible, but will clearly be suboptimal):

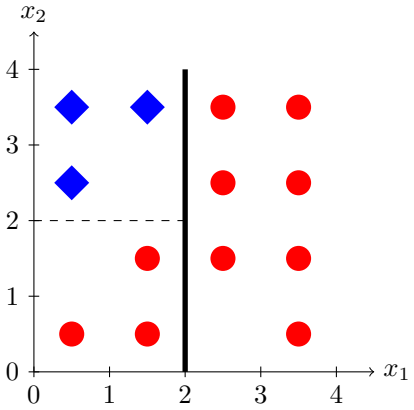


The corresponding cost function will take the following values for the different splits:

Split	n_1	$\widehat{\pi}_{1B}$	$\widehat{\pi}_{1R}$	$Q_1(T)$	n_2	$\widehat{\pi}_{2B}$	$\widehat{\pi}_{2R}$	$Q_2(T)$	n_3	$\widehat{\pi}_{3B}$	$\widehat{\pi}_{3R}$	$Q_3(T)$	$n_1Q_1(T)+n_2Q_2(T)+n_3Q_3(T)$
$x_2 < 2$ ($x_1 \leq 1$)	2	2/2	0/2	0	1	0/1	1/1	0	10	1/10	9/10	1/10	1
$x_2 < 3$ ($x_1 > 1$)	3	1/3	2/3	1/3	3	1/3	2/3	1/3	7	0/7	7/7	0	2
$x_1 < 2$ ($x_1 > 1$)	3	1/3	2/3	1/3	3	1/3	2/3	1/3	7	0/7	7/7	0	2

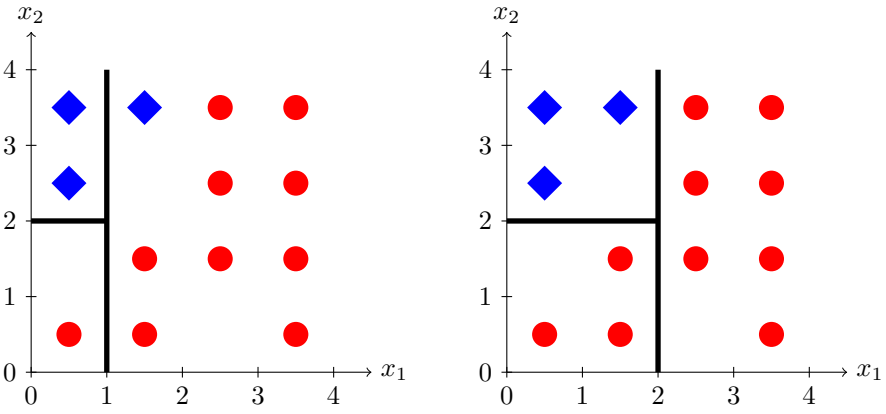
Thus, the optimal decision (with respect to the misclassification rate) is to make the split at $x < 2 = 2$ in the $x_1 < 1$-region.

(ii) For the Gini-index based tree, the only sensible split to make is:



The cost function with the Gini-index is 0 with a split at $x_2 = 2$, and it is thus optimal.

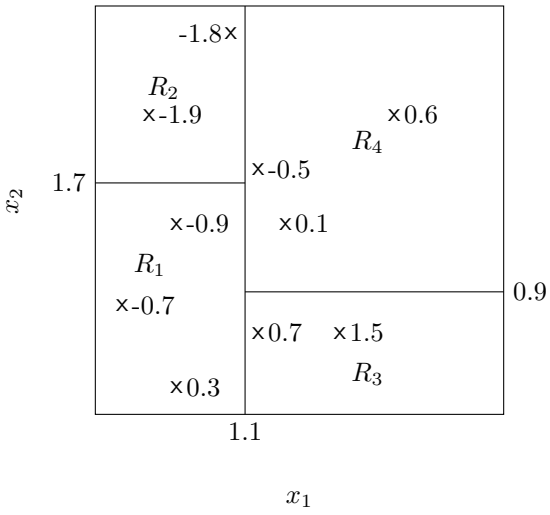
The trees obtained after two splittings are thus



Clearly, the right tree (obtained by the Gini-index based cost function) is preferred over the left (obtained by the cost function based on the classification error rate). The reason is that the sequential splitting algorithm is 'greedy' or 'myopic', by making sequentially decisions only looking one step ahead. The Gini index, however, counteracts

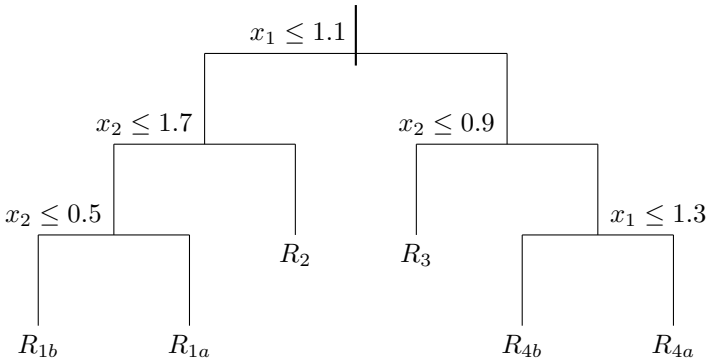
this by promoting splits that gives one relatively pure region (few misclassifications) and another more mixed region (more misclassifications), which later can be split again.

7.4 (a) The partitioning of the input space is as follows

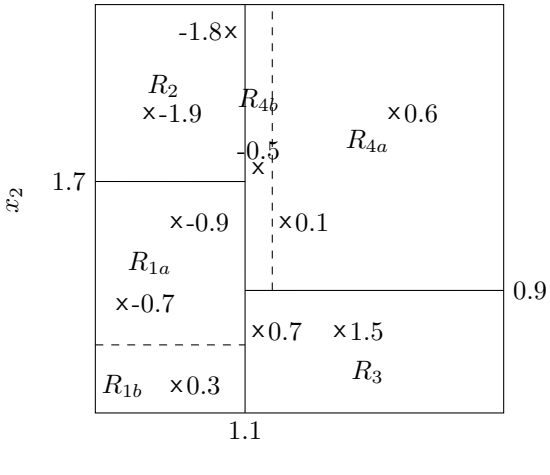


(b) Since $x_{\star 1} = 1.5 > 1.1$ and $x_{\star 2} = 1.8 > 0.9$, the test point belongs to region R_4. To compute the predicted output we also need to know which regions the training data points fall into. This is also shown in the figure above, from which it is clear that data points $i = 1, 2, 8$ are in R_4. The mean of these points is $\widehat{y}_{R_4} = 0.07$, which thus becomes our prediction $\widehat{y}_\star$.

(c) Regions R_2 and R_3 already have two or fewer data points, whereas region R_1 and R_4 need another split, respectively. We find that the RSS-minimizing split of R_1 is for somewhere between $x_2 = 0.2$ and $x_2 = 0.8$, and for R_4 somewhere between $x_1 = 1.2$ and $x_1 = 1.4$, as for example



This deeper tree thus corresponds to a partitioning which looks like



(d) $\mathbf{x}_\star$ now belongs to R_{4a}, which gives $\widehat{y}_\star = \widehat{y}_{R_{4a}} = 0.35$.