ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

HS 2023

Prof. R. Wattenhofer
Benjamin Estermann

Computational Thinking
Sample Solutions to Exercise 10

1 Global Minimum

You want to find the global minimum of f using gradient descent, where $f = 3x^4 - 4x^3 - 12x^2 + 4$

a) The extrema of f can be found by differentiating:

$$f' = 12(x^3 - x^2 - 2x) = 12x(x-2)(x+1)$$

Therefore the extrema are at $x = -1, 0, 2$. The function values at these points are $-1, 4, -28$, and the coefficient of x^4 is positive, so the global minimum is at $x = 2$. With an appropriate learning rate, gradient descent will converge to $x = 2$ for all $x_0 > 0$, since $x = 0$ is the nearest maximum and there are no maxima in the other direction. See Figure 1 below.

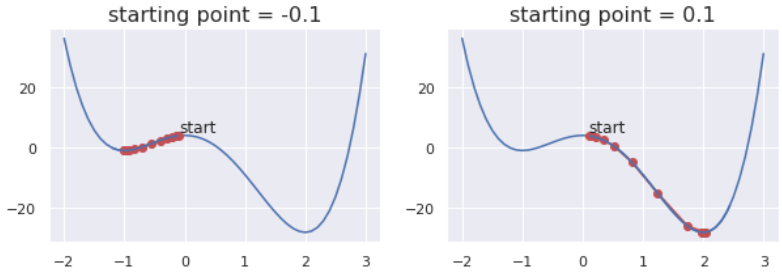

Figure 1: Convergence of gradient descent for different starting values with $f = 3x^4 - 4x^3 - 12x^2 + 4$.

b) We know that the global minimum is $x = 2$ and the gradient at $x = 3$ is

$$f' = 12x(x-2)(x+1) = 144$$

So to reach the global minimum directly in a single step we must have

$$\begin{aligned} 2 &= 3 - 144\alpha \\ \implies \alpha &= \frac{1}{144} \end{aligned}$$

c) The learning rate for Newton's method is 1 over the second derivative, which is

$$\begin{aligned} f''(x) &= 12(3x^2 - 2x - 2) \\ \implies f''(3) &= 12(19) = 228 \end{aligned}$$

This does not give the optimal learning rate. This is not surprising, because the Newton method is a second order method (based on the second order Taylor expansion), so optimality is only guaranteed for second order functions.

d) On the other hand $f = ax^2 + bx + c$ is a second order function, so the Newton method will give the optimal learning rate and the global minimum will be found in a single step, whatever the starting point. If you are not convinced, try it out ;)

$$\begin{aligned} f &= ax^2 + bx + c \\ f' &= 2ax + b \\ f'' &= 2a \end{aligned}$$

So the minimum is at

$$2ax + b = 0 \implies x = \frac{-b}{2a}$$

And starting at x_0, after one step we are at

$$\begin{aligned} x_1 &= x_0 - \frac{1}{f''(x_0)} \left(f'(x_0)\right) \\ &= x_0 - \frac{1}{2a} \left(2ax_0 + b\right) = \frac{-b}{2a} \end{aligned}$$

2 Logistic Regression & XOR

We want to learn the "XOR" function with logistic regression. Our input space is $\mathcal{X} = \{0, 1\}^2$ and our output space is $\mathcal{Y} = \{0, 1\}$ and we want to learn the mapping

$$(x_1, x_2) \mapsto x_1 \oplus x_2$$

a) Logistic regression can only learn linear decision boundaries, but the classes of "XOR" can not be separated by a single line/hyperplane.

b) Logistic regression can learn "XOR" by manually adding features. You can for example add $x_1 x_2$ as a feature. Consider the function

$$\hat{f} = \psi(\boldsymbol{w}^T \boldsymbol{x}) = \psi(-1 + 2x_1 + 2x_2 - 4x_1 x_2)$$

where ψ is the logistic function, $\boldsymbol{x} = \{1, x_1, x_2, x_1 x_2\}^T$, and $\boldsymbol{w} = \{-1, 2, 2, -4\}^T$. Then we have the following values for $\hat{f}$ in Table 1

x_1	x_2	$x_1 x_2$	$\boldsymbol{w}^T \boldsymbol{x}$	$\hat{f}$	y
1	1	1	-1	0.27	0
1	0	0	1	0.73	1
0	1	0	1	0.73	1
0	0	0	-1	0.27	0

Table 1: Logistic Regression for "XOR" with additional feature $x_1 x_2$

c) How about "AND", "OR", "NOT AND"? Can logistic regression learn these?

Yes, we show the separating hyperplanes in Figure 2 below.

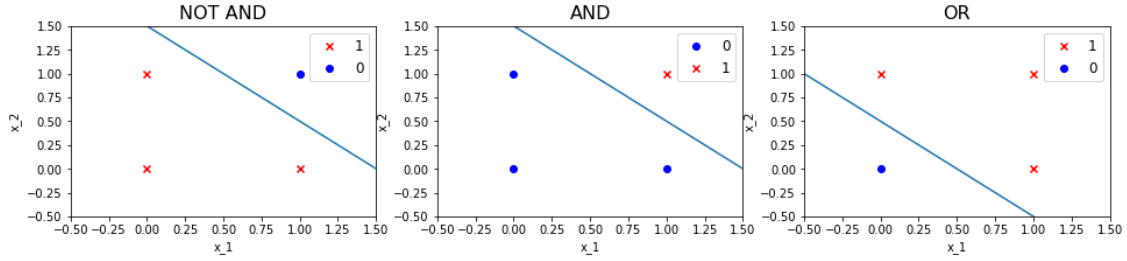

Figure 2: Logistic regression decision boundaries for binary logic

d) Show that "hierarchical" logistic regression with 2 layers can learn "XOR". What does this remind you of?

We combine the "AND", "OR", "NOT AND" gates from the previous question. A possible solution with weights along the edges is shown in Figure 3 below. This is similar to a very simple neural network.

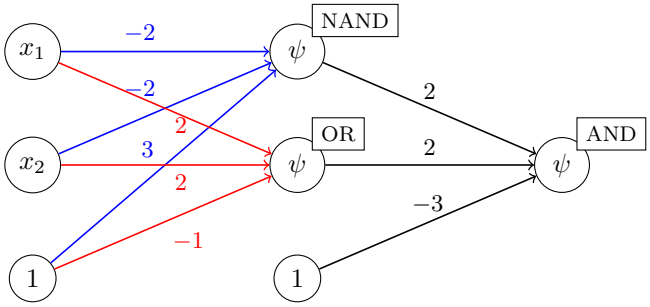

Figure 3: Hierarchical Logistic Regression for "XOR"

e) A decision tree can learn "XOR"; see example decision boundaries in Figure 4 below. However, depending on the loss function and the stopping criterion, we might not make the first split. This split still leaves perfectly balanced subsets and might not even improve the total loss (depending on the chosen loss function).

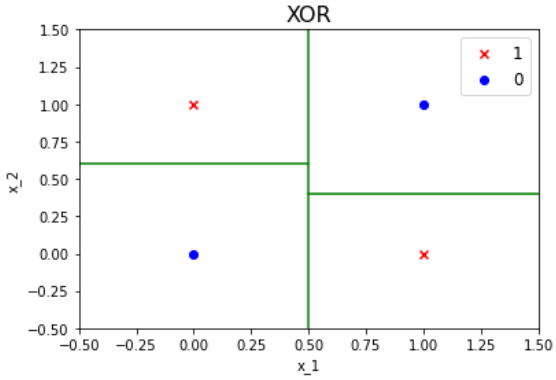

Figure 4: Example decision tree boundary for "XOR"

3 Gini Impurity

a) Construct an optimal decision tree (requiring the minimum number of splits). Note that decision trees only make axis-parallel splits, so a single split along the diagonal is not possible.

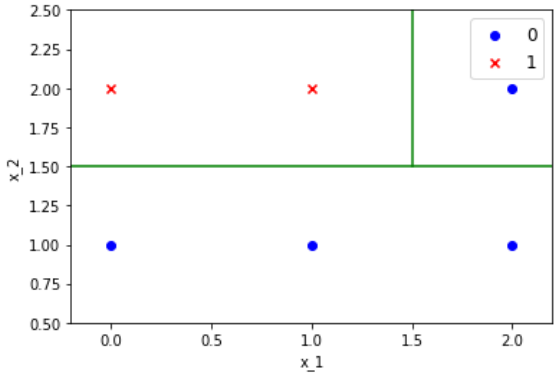

Figure 5: Example dataset separated with minimum number of splits.

b) It is fairly clear what the best splits for "Gini" are, but we show some calculations to develop familiarity with this splitting criterion.

As a first split only 2 options really come into question: $(i = 1, 1.5)$ and $(i = 2, 1.5)$

$$\begin{aligned} L(1, 1.5) &= \frac{4}{6}\left(1 - (p_0^1)^2 - (p_1^1)^2\right) + \frac{2}{6}\left(1 - (p_0^2)^2 - (p_1^2)^2\right) \\ &= \frac{4}{6}\left(1 - \frac{1}{2}^2 - \frac{1}{2}^2\right) + \frac{2}{6}(1 - 1 - 0) \\ &= \frac{2}{3}\left(\frac{1}{2}\right) = \frac{1}{3} \\ L(2, 1.5) &= \frac{3}{6}\left(1 - \frac{1}{3}^2 - \frac{2}{3}^2\right) + \frac{3}{6}(0) \\ &= \frac{2}{9} < \frac{1}{3} \end{aligned}$$

where p_0^1 denotes the proportion of negative samples on one side of the split with respect to the total samples on that side, whereas p_0^2 denotes the proportion of negative samples on the other side.

Therefore the first split is $(i = 2, 1.5)$. But the next split is clearly $(i = 1, 1.5)$.

c) See Figure 6 below for an example, where CART with Gini does not find an optimal decision tree, in terms of minimizing the number of splits.

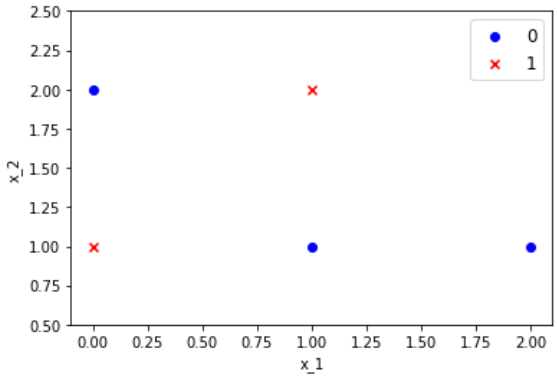

Figure 6: Example dataset, where CART with Gini does not find an optimal decision tree.

The problem here is that Gini prefers to split the points along $x_1 = 1.5$, rather than along $x_1 = 0.5$. Following this split, 3 more splits are required to perfectly separate the positive and negative instances. But using the split $x_1 = 0.5$, only 2 more splits are needed. The relevant calculations are shown below.

Substituting into the CART formula with the Gini loss, we can calculate the following losses for splits $(i = 1, 0.5)$ and $(i = 1, 1.5)$:

$$\begin{aligned} L(1, 0.5) &= \frac{2}{5}\left(1 - \frac{1}{2}^2 - \frac{1}{2}^2\right) + \frac{3}{5}\left(1 - \frac{2}{3}^2 - \frac{1}{3}^2\right) \\ &= \frac{2}{5}\left(\frac{1}{2}\right) + \frac{3}{5}\left(\frac{4}{9}\right) \\ &= \frac{42}{90} \\ L(1, 1.5) &= \frac{4}{5}\left(1 - \frac{1}{2}^2 - \frac{1}{2}^2\right) + \frac{1}{5}(0) \\ &= \frac{2}{5} < \frac{42}{90} \end{aligned}$$

By continuing the splitting, we get the decision boundaries shown in Figure 7 below.

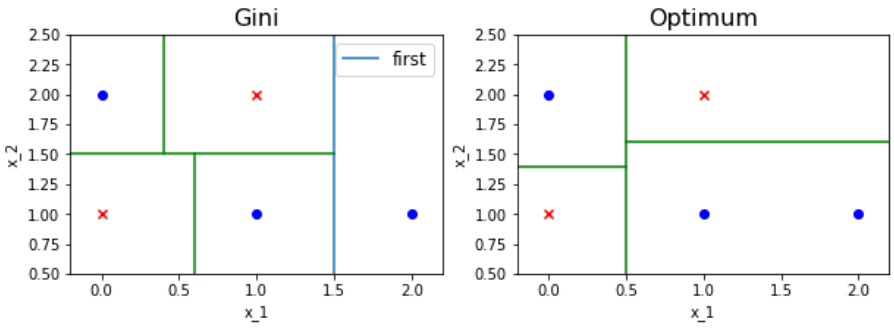

Figure 7: Example dataset, where CART with Gini does not find an optimal decision tree. Gini (left) uses 4 splits rather than the optimum (right) of 3