Regular Discussion 12

1 CalDining Bandits

You're an excited new student who wants to know where to eat lunch at Berkeley! Every day at lunchtime, you take action a to use your meal swipe at Crossroads ($a = X$), Cafe 3 ($a = C$), or Golden Bear Cafe ($a = G$) (the other dining halls are too inconvenient). Let a_i be the action you take on day i.

Suppose that the reward you get from croads (X) is uniformly distributed between -10 and 50, the reward you get from Cafe 3 (C) is uniformly distributed between 0 and 30, and the reward you get from GBC (G) is always 15.

(a) What is the optimal value V^*? Which dining hall has the best expected reward?

(b) What is the optimality gap Δ_C for the action of going to Cafe 3 (C)?

(c) Suppose Cafe 3 just happens to be right next to your dorm, so your policy is to always choose action C. What is the timestep regret under this policy?

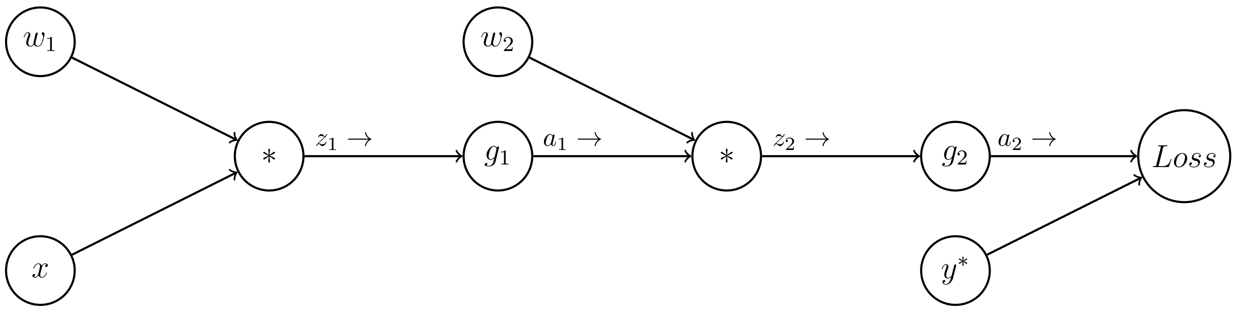
(d) Now suppose you are indecisive, so your policy is to randomly choose a dining hall to go to each day. What is the **regret** l_t for one action under this policy?

(e) Suppose you follow the random policy from the previous part for 5 days, taking actions X, C, C, G, X and getting rewards $10, 20, 22, 18, -10$. What is the **total regret** for this policy? (Hint: Trick question?)

(f) True or False: Using the UCB1 algorithm for this problem would lead to logarithmic total regret, after enough days.

2 Neural Nets

Consider the following computation graph for a simple neural network for binary classification. Here x is a single real-valued input feature with an associated class y^* (0 or 1). There are two weight parameters w_1 and w_2, and non-linearity functions g_1 and g_2 (to be defined later, below). The network will output a value a_2 between 0 and 1, representing the probability of being in class 1. We will be using a loss function $Loss$ (to be defined later, below), to compare the prediction a_2 with the true class y^*.



1. Perform the forward pass on this network, writing the output values for each node z_1, a_1, z_2 and a_2 in terms of the node's input values:

2. Compute the loss $Loss(a_2, y^*)$ in terms of the input x, weights w_i, and activation functions g_i:

3. Now we will work through parts of the backward pass, incrementally. Use the chain rule to derive $\frac{\partial Loss}{\partial w_2}$. Write your expression as a product of partial derivatives at each node: i.e. the partial derivative of the node's output with respect to its inputs. (Hint: the series of expressions you wrote in part 1 will be helpful; you may use any of those variables.)

4. Suppose the loss function is quadratic, $Loss(a_2, y^*) = \frac{1}{2}(a_2 - y^*)^2$, and g_1 and g_2 are both sigmoid functions $g(z) = \frac{1}{1+e^{-z}}$ (note: it's typically better to use a different type of loss, *cross-entropy*, for classification problems, but we'll use this to make the math easier).

   Using the chain rule from Part 3, and the fact that $\frac{\partial g(z)}{\partial z} = g(z)(1 - g(z))$ for the sigmoid function, write $\frac{\partial Loss}{\partial w_2}$ in terms of the values from the forward pass, y^*, a_1, and a_2:

5. Now use the chain rule to derive $\frac{\partial Loss}{\partial w_1}$ as a product of partial derivatives at each node used in the chain rule:

6. Finally, write $\frac{\partial Loss}{\partial w_1}$ in terms of x, y^*, w_i, a_i, z_i:

7. What is the gradient descent update for w_1 with step-size α in terms of the values computed above?