CS 188
Spring 2024

Regular Discussion 12 Solutions

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?

$V^* = \text{argmax}_a E(r|a) = \boxed{20}$

The best action is to go to croads (hot take).

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

$Q(C) = E(r|C) = 15$

$\Delta_C = V^* - Q(C) = 5$

(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?

$l_t = E[V^* - Q(a_t)] = V^* - Q(C) = 5$

(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?

$l_t = E[V^* - Q(a_t)]$

$= \frac{1}{3}(V^* - Q(X)) + \frac{1}{3}(V^* - Q(C)) + \frac{1}{3}(V^* - Q(G))$

$= 0 + \frac{5}{3} + \frac{5}{3}$

$= \boxed{\frac{10}{3}}$

(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?)

In this class, regret is used to refer to "expected suboptimality", and total regret is also an expectation. As such, the total regret is 5 times the result from the previous part, so

$L_5 = \boxed{\frac{50}{3}}$

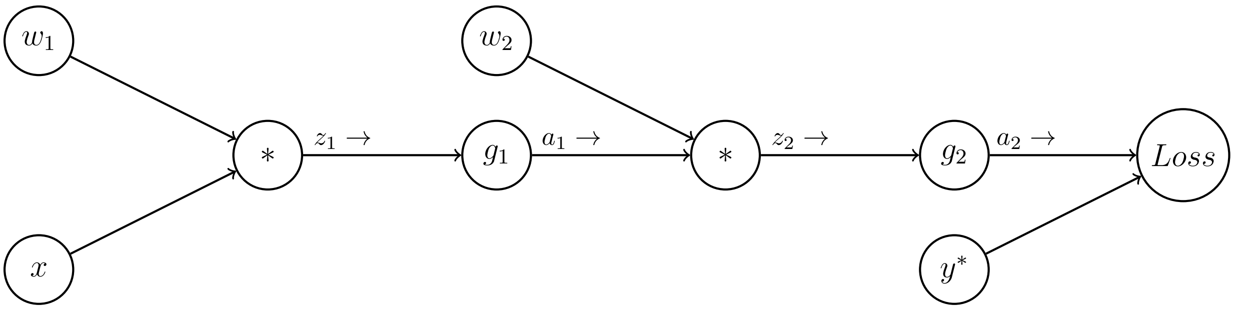
Note that total regret doesn't always have to be a linear multiplication of the regret for one step! If your policy changes with time/new observations, your regret at each step might change as time goes on. For example, using the UCB1 algorithm leads to logarithmic total regret.

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

True, taken directly from lecture slides.

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:

$$z_1 = x * w_1$$
$$a_1 = g_1(z_1)$$
$$z_2 = a_1 * w_2$$
$$a_2 = g_2(z_2)$$

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

   Recursively substituting the values computed above, we have:

$$Loss(a_2, y^*) = Loss(g_2(w_2 * g_1(w_1 * x)), y^*)$$

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.)

$$\frac{\partial Loss}{\partial w_2} = \frac{\partial Loss}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial w_2}$$

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:

   First we'll compute the partial derivatives at each node:

$$\frac{\partial Loss}{\partial a_2} = (a_2 - y^*)$$
$$\frac{\partial a_2}{\partial z_2} = \frac{\partial g_2(z_2)}{\partial z_2} = g_2(z_2)(1 - g_2(z_2)) = a_2(1 - a_2)$$
$$\frac{\partial z_2}{\partial w_2} = a_1$$

   Now we can plug into the chain rule from part 3:

$$\frac{\partial Loss}{\partial w_2} = \frac{\partial Loss}{\partial a_2}\frac{\partial a_2}{\partial z_2}\frac{\partial z_2}{\partial w_2} = (a_2 - y^*) * a_2(1 - a_2) * a_1$$

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:

$$\frac{\partial Loss}{\partial w_1} = \frac{\partial Loss}{\partial a_2}\frac{\partial a_2}{\partial z_2}\frac{\partial z_2}{\partial a_1}\frac{\partial a_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

6. Finally, write $\frac{\partial Loss}{\partial w_1}$ in terms of x, y^*, w_i, a_i, z_i: The partial derivatives at each node (in addition to the ones we computed in Part 4) are:

$$\frac{\partial z_2}{\partial a_1} = w_2$$
$$\frac{\partial a_1}{\partial z_1} = \frac{\partial g_1(z_1)}{\partial z_1} = g_1(z_1)(1 - g_1(z_1)) = a_1(1 - a_1)$$
$$\frac{\partial z_1}{\partial a_1} = x$$

   Plugging into the chain rule from Part 5 gives:

$$\frac{\partial Loss}{\partial w_1} = \frac{\partial Loss}{\partial a_2}\frac{\partial a_2}{\partial z_2}\frac{\partial z_2}{\partial a_1}\frac{\partial a_1}{\partial z_1}\frac{\partial z_1}{\partial w_1} = (a_2 - y^*) * a_2(1 - a_2) * w_2 * a_1(1 - a_1) * x$$

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

$$w_1 \leftarrow w_1 - \alpha(a_2 - y^*) * a_2(1 - a_2) * w_2 * a_1(1 - a_1) * x$$