1 Optimization

We would like to classify some data. We have $N$ samples, where each sample consists of a feature vector $\mathbf{x} = [x_1, \cdots, x_k]^T$ and a label $y \in \{0, 1\}$.

Logistic regression produces predictions as follows:

$$P(Y = 1 \mid \mathbf{X}) = h(\mathbf{x}) = s \left( \sum_i w_i x_i \right) = \frac{1}{1 + \exp(-\sum_i w_i x_i)}$$

$$s(\gamma) = \frac{1}{1 + \exp(-\gamma)}$$

where $s(\gamma)$ is the logistic function, $\exp x = e^x$, and $\mathbf{w} = [w_1, \cdots, w_k]^T$ are the learned weights.

Let’s find the weights $w_j$ for logistic regression using stochastic gradient descent. We would like to minimize the following loss function (called the cross-entropy loss) for each sample:

$$L = -[y \ln h(\mathbf{x}) + (1 - y) \ln(1 - h(\mathbf{x}))]$$

(a) Show that $s'(\gamma) = s(\gamma)(1 - s(\gamma))$

(b) Find $\frac{dL}{dw_j}$. Use the fact from the previous part.

(c) Now, find a simple expression for $\nabla_{\mathbf{w}} L = [\frac{dL}{dw_1}, \frac{dL}{dw_2}, \ldots, \frac{dL}{dw_k}]^T$

(d) Write the stochastic gradient descent update for $\mathbf{w}$. Our step size is $\eta$. 

\[ \mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} L \]
Q2. RL

Pacman is in an unknown MDP where there are three states [A, B, C] and two actions [Stop, Go]. We are given the following samples generated from taking actions in the unknown MDP. For the following problems, assume \( \gamma = 1 \) and \( \alpha = 0.5 \).

(a) We run Q-learning on the following samples:

<table>
<thead>
<tr>
<th>s</th>
<th>a</th>
<th>s'</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Go</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>Stop</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>Stop</td>
<td>A</td>
<td>-2</td>
</tr>
<tr>
<td>B</td>
<td>Go</td>
<td>C</td>
<td>-6</td>
</tr>
<tr>
<td>C</td>
<td>Go</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>A</td>
<td>Go</td>
<td>A</td>
<td>-2</td>
</tr>
</tbody>
</table>

What are the estimates for the following Q-values as obtained by Q-learning? All Q-values are initialized to 0.

(i) \( Q(C, \text{Stop}) = \) __________________________

(ii) \( Q(C, \text{Go}) = \) __________________________

(b) For this next part, we will switch to a feature based representation. We will use two features:

- \( f_1(s, a) = 1 \)
- \( f_2(s, a) = \begin{cases} 1 & a = \text{Go} \\ -1 & a = \text{Stop} \end{cases} \)

Starting from initial weights of 0, compute the updated weights after observing the following samples:

<table>
<thead>
<tr>
<th>s</th>
<th>a</th>
<th>s'</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Go</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>Stop</td>
<td>A</td>
<td>0</td>
</tr>
</tbody>
</table>

What are the weights after the first update? (using the first sample)

(i) \( w_1 = \) __________________________

(ii) \( w_2 = \) __________________________

What are the weights after the second update? (using the second sample)

(iii) \( w_1 = \) __________________________

(iv) \( w_2 = \) __________________________
3 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. [Optional] 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. [Optional] 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 $\frac{\partial L_{loss}}{\partial w_2}$ in terms of the values from the forward pass, $y^*$, $a_1$, and $a_2$:

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

6. [Optional] Finally, write $\frac{\partial L_{loss}}{\partial w_1}$ in terms of $x$, $y^*$, $w_i$, $a_i$, $z_i$:

7. [Optional] What is the gradient descent update for $w_1$ with step-size $\alpha$ in terms of the values computed above?