CS547 Homework 3

Anthony Marcozzi

October 21, 2021

Problem 1: Overparameterization

With a 3 class classifier note that $N = 3$, and our parameter matrix w looks like

$$\begin{bmatrix} w_{0,0} & w_{0,1} \\ w_{1,0} & w_{1,1} \\ w_{2,0} & w_{2,1} \end{bmatrix}$$

where each row of the matrix contains a vector of parameters for each of the N classes. Our goal is to represent the same model, but with $2(N-1) = 4$ parameters instead of our current $2N = 6$ parameters.

Proof.

First, I make the observation that a logistic regression function with one feature uses two parameters w_0 and w_1

$$P(y = k|x) = \sigma(\phi\vec{w}) = \frac{1}{1 + \exp(-\phi\vec{w})} \tag{1}$$

So with this observation in mind, and the hint from the problem statement about recovering logistic regression, consider the following approach. Use two logistic regression functions: one representing the log odds of class one and the second representing the log odds of class two. Notice that this only considers the unscaled log probabilities of classes one and two. Thus, to find the unscaled log probability of class three we can take the complement of the log probability of class one or two. All together, the model looks like

$$P(y = 1|x) = \sigma(\phi\vec{w_1}) = \frac{1}{1 + \exp(-\phi\vec{w_1})} \tag{2}$$

$$P(y = 2|x) = \sigma(\phi\vec{w_2}) = \frac{1}{1 + \exp(-\phi\vec{w_2})} \tag{3}$$

$$P(y = 3|x) = 1 - (\sigma(\phi\vec{w_1}) + \sigma(\phi\vec{w_2})) \tag{4}$$

Finally, to find any $P(y = k|x)$ for $k \in 1, 2, 3$ we can write

$$P(y = k|x) = \max_k [P(y = 1|x), P(y = 2|x), P(y = 3|x)] \tag{5}$$

Where each $P(y = k|x)$ is defined in equations 2-5, and our parameters are $\vec{w_1} = [w_0, w_1]$ and $\vec{w_2} = [w_0, w_1]$, meeting our criteria for three classes.

□

Please see the attached IPython notebook for a valid implementation of this technique on the three-class Iris classification problem.

Problem 2: Backpropogation

A

Proof. Derive and evaluate an expression for $\frac{\partial L}{\partial w_2^{(2)}}$

Using equations (2)-(9) listed on Page 1 of the Homework assignment, we can apply the chain rule to the partial derivative $\frac{\partial L}{\partial w_2^{(2)}}$ to get

$$\frac{\partial L}{\partial w_2^{(2)}} = \frac{\partial L}{\partial z^3} \frac{\partial z^3}{\partial a^{(3)}} \frac{\partial a^{(3)}}{\partial z_2^{(2)}} \frac{\partial z_2^{(2)}}{\partial a_2^{(2)}} \frac{\partial a_2^{(2)}}{\partial w_2^{(2)}} \tag{6}$$

Expanding and solving for each of these partial derivatives we get the following system of equations

$$L = \frac{1}{2}(z^{(3)} - y)^2 \Rightarrow \frac{\partial L}{\partial w_2^{(2)}} = (z^{(3)} - y) \tag{7}$$

$$z^{(3)} = a^{(3)} \Rightarrow \frac{\partial z^3}{\partial a^{(3)}} = 1 \tag{8}$$

$$a^{(3)} = z_1^{(2)} w_1^{(3)} + z_2^{(2)} w_2^{(3)} \Rightarrow \frac{\partial a^{(3)}}{\partial z_2^{(2)}} = w_2^3 \tag{9}$$

$$z_2^{(2)} = \sigma(a_2^{(2)}) \Rightarrow \frac{\partial z_2^{(2)}}{\partial a_2^{(2)}} = \sigma'(a_2^{(2)}) \tag{10}$$

$$a_2^{(2)} = z^{(1)} w_2^{(2)} \Rightarrow \frac{\partial a_2^{(2)}}{\partial w_2^{(2)}} = z^{(1)} \tag{11}$$

Thus, by combining equation (6) with equations (7)-(11) we can find an analytic solution for the partial derivative

$$\frac{\partial L}{\partial w_2^{(2)}} = (z^{(3)} - y) \cdot w_2^3 \cdot \sigma'(a_2^{(2)}) \cdot z^{(1)} \tag{12}$$

Please see the attached IPython notebook for verification of this calculation.

□

B

While the Finite Difference Method provides a convenient method for finding the derivatives of expressions that are difficult to solve, or that do not have an analytic solution, this method is not appropriate for this case for several reasons. Firstly, the results of the Finite Difference Method are sensitive to values of epsilon, the parameter that stands for step size in the derivative. Thus, we add an additional ϵ parameter for each of our model parameters with this technique. Secondly, solving the partial derivative analytically takes constant computational time, whereas the Finite Difference Method is linear with respect to step size and tolerance in its computation. Finally, it simply doesn't make sense to approximate a derivative when we can find the exact derivative using an analytic solution.

Problem 3: Robust Cross-Entropy

Find the negative log likelihood of a binary classification problem with some probability, ϵ, of mislabeled class labels.

Proof.

Begin by applying the Sum and Product rules to $P(y_{obs}|\vec{x})$

$$P(y_{obs}|\vec{x}) = \sum_k P(y_{obs}, y = k|\vec{x}) = \sum_k P(y_{obs}|y = k)P(y = k|\vec{x}) \tag{13}$$

Since there are only two classes, 0 and 1, write out the full expression for $k \in [0, 1]$

$$P(y_{obs}|\vec{x}) = P(y_{obs}|y = 1)P(y = 1|\vec{x}) + P(y_{obs}|y = 0)P(y = 0|\vec{x}) \tag{14}$$

Note that $P(y = k|\vec{x})$ is simply logistic regression, so we can write $P(y = k|\vec{x}) = \sigma(\phi\vec{x})$. Without loss of generality, suppose that $P(y = 1|\vec{x}) = \sigma(\phi\vec{x})$ and therefore $P(y = 0|\vec{x}) = (1 - \sigma(\phi\vec{x}))$.

Next, consider the two remaining $P(y_{obs}|y = k)$ terms. This term represents the probability of our model being labeled incorrectly. Because there are two possible events: a correct label or an incorrect label, we can represent the probability of an incorrect label occurring with ϵ. The intention of this term is to modulate the logit term with a coefficient $(1-\epsilon$ that represents the confidence in our log probability. Thus, to achieve this effect consider the case where $y_k = 1$. If $y_{obs} = 1$, then the label is correctly identified and we want to multiply the σ term by 1. However, if $y_{obs} = 0$, then we want to multiply the σ term by our confidence in a correct label $(1 - \epsilon)$. The logic for the other case, $y_k = 0$ is the same. We can describe this relationship with the following expressions

$$P(y_{obs}|y = 1) = (1 - \epsilon)^{y_{obs}}\epsilon^{1-y_{obs}} \tag{15}$$

$$P(y_{obs}|y = 0) = \epsilon^{y_{obs}}(1 - \epsilon)^{1-y_{obs}} \tag{16}$$

Putting all of these terms together we get the following sum

$$P(y_{obs}|\vec{x}) = (1 - \epsilon)^{y_{obs}}\epsilon^{1-y_{obs}}\sigma(\phi\vec{x}) + \epsilon^{y_{obs}}(1 - \epsilon)^{1-y_{obs}}(1 - \sigma(\phi\vec{x})) \tag{17}$$

Next, find the negative log likelihood of this expression

$$L = -\ln((1 - \epsilon)^{y_{obs}}\epsilon^{1-y_{obs}}\sigma(\phi\vec{x}) + \epsilon^{y_{obs}}(1 - \epsilon)^{1-y_{obs}}(1 - \sigma(\phi\vec{x}))) \tag{18}$$

□