Solutions

Part I – Logistic regression backpropagation with a single training example

In this part, you are using the Stochastic Gradient Optimizer to train your Logistic Regression. Consequently, the gradients leading to the parameter updates are computed on a single training example.

a) Forward propagation equations

Before getting into the details of backpropagation, let’s spend a few minutes on the forward pass. For one training example \( x = (x_1, x_2, ..., x_n) \) of dimension \( n \), the forward propagation is:

\[
\begin{align*}
    z &= wx + b \\
    \hat{y} &= a = \sigma(z) \\
    L &= -(y \log(\hat{y}) + (1-y) \log(1-\hat{y}))
\end{align*}
\]

b) Dimensions of the variables in the forward propagation equations

It’s important to note the shapes of the quantities in the previous equations:

\( x = (n, 1), \ w = (1, n), \ b = (1, 1), \ z = (1, 1), \ a = (1, 1), \ L \ is \ a \ scalar. \)

c) Backpropagation equations

Training our model means updating our weights and biases, \( W \) and \( b \), using the gradient of the loss with respect to these parameters. At every step, we need to calculate:

\[
\begin{align*}
    \frac{\partial L}{\partial w} &= \frac{\partial L}{\partial a} \\
    \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial a}
\end{align*}
\]

To do this, we will apply the chain rule.

\[
\begin{align*}
    \frac{\partial L}{\partial w} &= \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial w} \\
    \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial b}
\end{align*}
\]

So we need to calculate the following derivatives:
We will calculate those derivatives to get an expression of \( \frac{\partial L}{\partial w} \) and \( \frac{\partial L}{\partial b} \).

\[
\frac{\partial L}{\partial a} = -(y \frac{\partial \log(a)}{\partial a} + (1 - y) \frac{\partial \log(1 - a)}{\partial a}) \\
= -(y \frac{1}{a} + (1 - y) \frac{1}{1 - a}(-1))
\]

\[
\frac{\partial L}{\partial z} = -(y \frac{1}{a}(1 - a) + (1 - y) \frac{1}{a - 1}(-1)a(1 - a)) \\
= -(y \frac{1}{a}(1 - a) + (1 - y) \frac{1}{a - 1}(-1)a(1 - a)) \\
= -y(1 - a) - a(1 - y) \\
= a - y
\]

\[
\frac{\partial L}{\partial w} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial w} = (a - y)X^T
\]

Why did we choose \( X.T \) rather than \( X \)? We can have a look at the following dimensions without forgetting that the dimensions of the derivative of a term are the same as the dimensions of the term.

\[
\frac{\partial L}{\partial w} \quad \frac{\partial z}{\partial w} \quad a - y \quad X^T \\
(1, n) \quad (1, n) \quad (1, 1) \quad (1, n)
\]

\[
\frac{\partial L}{\partial b} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial b} = (a - y).1
\]

Then:

\[
w = w - \alpha(a - y)X^T
\]

\[
b = b - \alpha(a - y).1
\]
Part II - Backpropagation for a batch of $m$ training examples

In this part, you are using a Batch Gradient Optimization to train your Logistic Regression. Consequently, the gradients leading to the parameter updates are computed on the entire batch of $m$ training examples.

a) Write down the forward propagation equations leading to $J$.

b) Analyze the dimensions of all the variables in your forward propagation equations.

c) Write down the backpropagation equations to compute $\frac{\partial J}{\partial w}$.

a) Forward propagation equations

Before getting into the details of backpropagation, let’s study the forward pass.
For a batch of $m$ training examples, each of dimension $n$, the forward propagation is:

$$ z = wX + b \quad (1) $$
$$ a = \sigma(z) \quad (2) $$

$$ J = \sum_{i=1}^{m} L^{(i)} $$

where $L$ is the binary cross entropy loss

$$ L^{(i)} = y^{(i)} \log(a^{(i)}) + (1 - y^{(i)}) \log(1 - a^{(i)}) $$

b) Dimensions of the variables in the forward propagation equations

It's important to note the shapes of the quantities in equations (1) and (2).

$$ w = \mathbb{R}^{1 \times n}, \quad X = \mathbb{R}^{n \times m}, \quad b = \mathbb{R}^{1 \times m} $$

but is really of shape $1 \times 1$ and broadcasted to $1 \times m$

$$ z = \mathbb{R}^{1 \times m} \quad \text{and} \quad a = \mathbb{R}^{1 \times m} \quad \text{and} \quad J \text{ is a scalar.} $$

c) Backpropagation equations

To train our model, we need to update our weights and biases $w$ and $b$, using the gradient of the loss with respect to these parameters. In other words, we need to calculate

$$ \frac{\partial J}{\partial w} \text{ and } \frac{\partial J}{\partial b}. $$

To do this, we will apply the chain rule.

We can write $\frac{\partial J}{\partial w}$ as $\frac{\partial J}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w}$

The first step is to calculate $\frac{\partial J}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w}$. 
\[
\frac{\partial J}{\partial a} = \sum_{i=1}^{m} \frac{\partial L^{(i)}}{\partial a^{(i)}} = - \sum_{i=1}^{m} \frac{\partial}{\partial a^{(i)}} \left[ y^{(i)} \log(a^{(i)}) + (1 - y^{(i)}) \log(1 - a^{(i)}) \right] = - \sum_{i=1}^{m} \left( \frac{y^{(i)}}{a^{(i)}} + (1 - y^{(i)}) \frac{1}{1-a^{(i)}} \right)
\]
and
\[
\frac{\partial L^{(i)}}{\partial z^{(i)}} = a^{(i)}(1 - a^{(i)}) \text{ which is the derivative of the sigmoid function.}
\]

Putting this together,
\[
\frac{\partial L}{\partial W} = \sum_{i=1}^{m} \frac{\partial L^{(i)}}{\partial W} \frac{\partial z^{(i)}}{\partial a^{(i)}}
\]
\[
\frac{\partial L^{(i)}}{\partial z^{(i)}} = \left( \frac{y^{(i)}}{a^{(i)}} - (1 - y^{(i)}) \frac{1}{1-a^{(i)}} \right) a^{(i)}(1 - a^{(i)}) = y^{(i)}(1 - a^{(i)}) + (1 - y^{(i)}) a^{(i)} = a^{(i)} - y^{(i)}
\]
\[
\frac{\partial z^{(i)}}{\partial w} = \frac{\partial}{\partial w} (wX_i + b) = \frac{\partial}{\partial w} wX_i = \frac{\partial}{\partial w} \sum_{j=0}^{n-1} w_j X_{ji}
\]

Therefore,
\[
\frac{\partial L}{\partial w} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)}) \frac{\partial z^{(i)}}{\partial w} \sum_{j=0}^{n-1} w_j X_{ji}
\]

To evaluate this derivative, we will find the derivative with respect to each element of \( W \).
\[
\frac{\partial L}{\partial w_p} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)}) \frac{\partial z^{(i)}}{\partial w_p} \sum_{j=0}^{n-1} w_{j} X_{ji}
\]
\[
\frac{\partial z^{(i)}}{\partial w_p} = \frac{\partial}{\partial w_p} (wX + b) = \frac{\partial}{\partial w_p} wX = \frac{\partial}{\partial w_p} \sum_{j=0}^{n-1} w_j X_{ji} = X_{pi} \text{ Where } X_p \text{ is a row vector corresponding to the } p^{th} \text{ row of the } X \text{ matrix.}
\]
\[
\frac{\partial L}{\partial w_p} = \sum_{i=1}^{m} (a^{(i)} - y^{(i)}) X_{pi}
\]

To get \( \frac{\partial L}{\partial w} \) we simply stack all these derivatives up, row wise.

This can efficiently be written in matrix form as:
\[
\frac{\partial L}{\partial w} = (A - Y)X^T
\]

Following a very similar procedure, and noting that \( \frac{\partial L}{\partial b} = 1 \)
\[
\frac{\partial L}{\partial w} = (A - Y).1 \text{ Where 1 is a column vector of 1's.}
\]

**Part III - Revisiting Backpropagation**

There are several possible ways to obtain an optimal set of weights/parameters for a neural network. The naive approach would consist in randomly generating a new set of weights at each iteration step. An improved method would use local information of the loss function (e.g. the gradient) to pick a better guess in the next iteration. Does backpropagation compute a numerical or analytical value of the gradients in a neural network? (Answer on Menti)
1. You are given the following neural network and your goal is to compute \( \frac{\partial L}{\partial a_1} \).

a) What other derivatives do you need to compute before finding \( \frac{\partial L}{\partial a_1} \)?

b) What values do you need to cache during the forward propagation in order to compute \( \frac{\partial L}{\partial a_1} \)?

A: a) You need to compute the intermediary derivatives \( \frac{\partial L}{\partial \hat{y}}, \frac{\partial \hat{y}}{\partial a_1}, \frac{\partial a_1^{[3]}}{\partial z_1^{[3]}}, \frac{\partial z_1^{[3]}}{\partial a_1}, \frac{\partial z_1^{[2]}}{\partial a_1^{[1]}}, \frac{\partial z_1^{[1]}}{\partial a_1^{[2]}} \).

b) \( d_z^{[L]} = da^{[L]} \odot g'(z^{[L]}) \)
\( dW^{[L]} = d_z^{[L]} \odot a^{[L-1]} \)
\( db^{[L]} = d_z^{[L]} \)
\( da^{[L-1]} = W^{[L]^T} d_z^{[L]} \)
\( d_z^{[L]} = W^{[L+1]^T} d_z^{[L+1]} \odot g'(z^{[L]}) \)
...

2. Backpropagation example on a univariate scalar function (e.g. \( f: R \rightarrow R \)):

Let’s suppose that you have built a model that uses the following loss function:

\[ L = (\hat{y} - y)^2 \] where \( \hat{y} = \tanh(\sigma(wx^2 + b)) \)

Assume that all the above variables are scalars. Using backpropagation, calculate \( \frac{\partial L}{\partial w} \).

A: \( \frac{\partial L}{\partial w} = 2(\hat{y} - y) \times \frac{\partial \hat{y}}{\partial w} = 2(\hat{y} - y) \times (1 - \hat{y}^2) \times \frac{\partial \hat{y}}{\partial w} = 2(\hat{y} - y) \times (1 - \hat{y}^2) \times z(1 - z) \times x^2 \)
where \( z = \sigma(wx^2 + b) \)
3. Backpropagation example on a multivariate scalar function (e.g. \( f : \mathbb{R}^n \to \mathbb{R} \)):

Let’s suppose that you have built a model that uses the following loss function:

\[
L = -y \log(\hat{y}) \quad \text{where} \quad \hat{y} = ReLU (w^T x + b)
\]

a) Assume \( x \in \mathbb{R}^n \). What’s the shape of \( w \)?

A: \( w \in \mathbb{R}^{n \times 1} \)

b) Using backpropagation, obtain \( \frac{\partial L}{\partial w} \).

A: We will derive \( \frac{\partial L}{\partial w_i} \) for i=1...n:

\[
\frac{\partial L}{\partial w_i} = -y \times \frac{\partial \log(\hat{y})}{\partial w_i} = -y \times \frac{1}{\hat{y}} \frac{\partial \hat{y}}{\partial w_i} = -y \times \frac{1}{\hat{y}} \times f(z) \times x_i \quad \text{where} \quad f(z) = 1, \quad \text{if} \quad z > 0, \quad f(z) = 0 \quad \text{otherwise} \quad \text{(where} \quad z = w^T x + b )
\]

4. Backpropagation applied to scalar-matrix functions (\( f : \mathbb{R}^{n \times m} \to \mathbb{R} \)):

The final case that is worth exploring is:

\[
L = \| \hat{y} - y \|^2 \quad \text{where} \quad \hat{y} = \sigma(x) \cdot W
\]

a) Assume that \( \hat{y} \in \mathbb{R}^{1 \times m} \) and \( x \in \mathbb{R}^{1 \times n} \). What is the shape of \( W \)?

A: \( W \) is an nxm matrix. (Note that the shapes of \( y \) and \( x \) differ from what you are used to in the class notations.)

b) Using backpropagation, calculate \( \frac{\partial L}{\partial x} \).

A: 

\[
\frac{\partial L}{\partial x} = 2(\hat{y} - y) \times \frac{\partial \hat{y}}{\partial W} = 2\hat{y} \times W^T \odot z \odot (1 - z) \quad \text{where} \quad z = \sigma(x)
\]