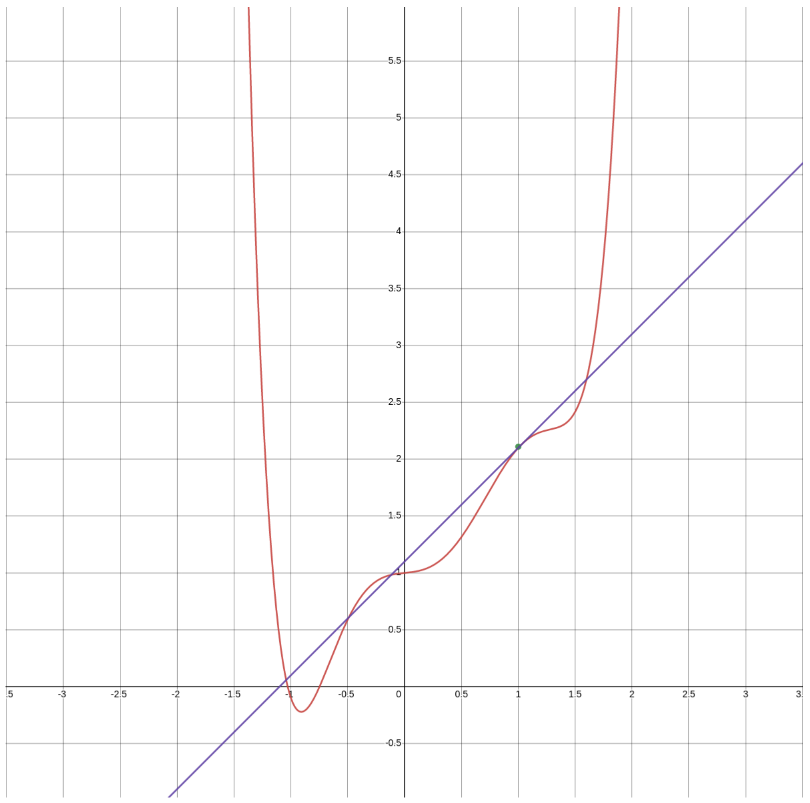
October 15, 2022
Homework 3
Sean Collings
#A02278024

1 Problem 1

1. Consider the following graph where the green dot represents our starting location, and the blue line represents the gradient (or derivative in this case) at that position. To begin gradient descent, we start moving in the *opposite* direction of that gradient, which moves us a bit down the "hill". We perform the operation many times to approach the minimum.



2. **Pro**: The network can be trained on each new piece of data as it is received. This is useful for processes in which labels are crowd-sourced. **Con**: Having a batch size of one will have a tendency to negatively affect speed of learning.

3. Code is included, uses PyTorch.

Accuracy:	98.27%
Learning Rate:	0.1
Hidden Neurons	1000

2 Problem 2

1. This is simply a substitution of element-wise vector multiplication with matrix multiplication. Consider this trivial example:

$$\nabla_a C = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \quad \sigma'(z^L) = \begin{bmatrix} 4 \\ 3 \\ 2 \\ 1 \end{bmatrix} \quad \Sigma'(z^L) = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then:

$$\nabla_a C \cdot \sigma'(z^L) = \begin{bmatrix} 4 \\ 6 \\ 6 \\ 4 \end{bmatrix} = \delta^L$$

By placing the the values of $\sigma'(z^L)$ on the diagonals of a matrix $\Sigma'(z^L)$ and flipping the order of operations, the output will be the same because all other terms cancel. Thus:

$$\Sigma'(z^L)\nabla_a C = \begin{bmatrix} 4 \\ 6 \\ 6 \\ 4 \end{bmatrix} = \delta^L$$

2. The exact same logic as the above problem can be applied here as well. By placing the the values of $\sigma'(z^L)$ on the diagonals of a matrix $\Sigma'(z^L)$ and flipping the order of operations, the output will be the same because all other terms cancel.

3. The definition of d^l is as follows:

$$\delta^l = \left((w^{l+1})^T \delta^{l+1}\right) \odot \sigma'(\mathbf{z}^L)$$

As shown in the previous problem this can be re-written as $\delta^l = \Sigma(z^l)(w^{l+1})^T \delta^{l+1}$. Because this is a recursive definition (written in terms of the d^{l+1} term) we can expand it as follows:

$$\delta^l = \Sigma(z^l)(w^{l+1})^T \times \Sigma(z^{l+1})(w^{l+2})^T \times \Sigma(z^{l+2})(w^{l+3})^T \times \cdots \times \delta^L$$

Where d^L is the final output of the network. And as a previous problem in this section showed, d^L can also be written as $\Sigma'(z^L)\nabla_a C$, thus:

$$\delta^l = \Sigma(z^l)(w^{l+1})^T \times \cdots \times \Sigma'(z^L)\nabla_a C$$

4. Backpropagation with Linear activation function. Note that because $a^l = \sigma(z^l) = z^l$, a^l has been replaced with z^l

   1. **Input** x: set the activation z^l for each layer

2. **Feedforward**: for each $l = 2, 3, \ldots, L$, compute $z^l = w^l z^{l-1} + b^l$
3. **Output Error** δ^L: compute $\delta^l = \nabla_{z^l} C$
4. **Backpropagate the Error**: For each $l = L-1, L-2, ..., 2$, compute $\delta^l = \big((w^{l+1})^T \delta^{l+1}\big)$
5. **Output**: The cost function gradient is $\frac{\partial C}{\partial w^l_{jk}} = z^{l-1}\delta^l_j$

3 Problem 3

1. -

2. The hessian of a convex function is PSD. We know from a previous homework that the sum of any two PSD matrices is PSD. So, for some function f and some function g: $\nabla^2 f + \nabla^2 g = \nabla^2(f+g)$. Because $\nabla^2 f$ and $\nabla^2 g$, must be PSD, so too $\nabla^2(f+g)$ must be PSD $\Rightarrow f + g$ must be convex.

3.

$$f(x) = \frac{1}{2}x^T A x + b^T x + c \tag{1}$$

$$\nabla f(x) = \begin{bmatrix} Ax_1 + b^T \\ Ax_2 + b^T \\ \vdots \\ Ax_d + b^T \end{bmatrix} \tag{2}$$

$$\nabla^2 f(x) = \begin{bmatrix} a_{1,1} & 0 & \ldots & 0 \\ 0 & a_{2,2} & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & \ldots & \ldots & a_{d,d} \end{bmatrix} \tag{3}$$

If we apply the definition of PD/PSD we find:

$$x^T \nabla^2 f x = \sum_{i=1}^{d} x_i^2 a_i$$

The parity of this summation is determined by the values of a_i because x_i^2 is always positive. When

$$\sum_{i=1}^{d} x_i^2 a_i \left\{ \begin{array}{lccl} > 0 & \Rightarrow & \text{PD} & \Rightarrow \text{Strictly Convex} \\ \geq 0 & \Rightarrow & \text{PSD} & \Rightarrow \text{Convex} \\ \text{else} & & & \Rightarrow \text{Non Convex} \end{array} \right\} \tag{4}$$

4. Consider

$$x = -1 \quad y = 0 \quad t = \tfrac{1}{2}$$

Apply these to the definition of convexity:

$$f\left(\frac{1}{2}(-1)+\left(1-\frac{1}{2}\right)0\right) \leq \frac{1}{2}f(-1)+\left(1-\frac{1}{2}\right)f(0) \tag{5}$$

$$f\left(-\frac{1}{2}\right) \leq \frac{1}{2}f(-1) \tag{6}$$

$$-\frac{1}{8} \not\leq -\frac{1}{2} \tag{7}$$

As the inequality does not hold, the function is not convex.

5. For $f(x)$ to be convex $f''(x)$ must be positive for all $x \in \mathbb{R}$. This can easily shown to be untrue:

$$f''(x) = 6x$$

$$f''(-1) = -6$$

6. $f''(x) = -\frac{1}{x^2}$. If $-f''(x) \geq 0$ for all $x > 0$, then f is concave. This is obviously true as $\frac{1}{x^2}$ will never take on a negative value.

7. Being both concave and convex means that f fulfills this equality

$$f(tx+(1-t)y) = tf(x)+(1-t)f(y)$$

Because this equality represents the overlap between the concave definition and convex definition. In essence, any function that fulfills both the concave and convex definition will also fulfill this.

We can apply this to $f(x)$

$$f(tx+(1-t)y) = tf(x)+(1-t)f(y) \tag{8}$$

$$a(tx+(1-t)y)+b = t(ax+b)+(1-t)(ay+b) \tag{9}$$

$$atx+ay-aty+b = atx+tb+ay+b-aty-tb \tag{10}$$

$$atx+ay-aty+b = atx+ay-aty+b \tag{11}$$

$$\tag{12}$$

These affine functions do *not* have a maximum of minimum. Because of this property, they are the only family of functions that satisfies the above equality and thus, are the only functions that are both convex and concave.

4 Problem 4

1. -

2.

λ	Error Rate
10	3.10%
1	2.70%
0.1	2.60%
0.01	2.60%

3. 20 most incorrectly labeled images