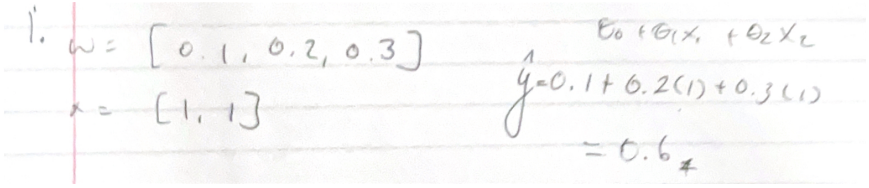
CS 577 - F22 - Assignment 2 Questions Portion (Part 1)

Due date: 10/04/2022

Anas Puthawala - A20416308
Professor Gady Agam

Artificial Neurons

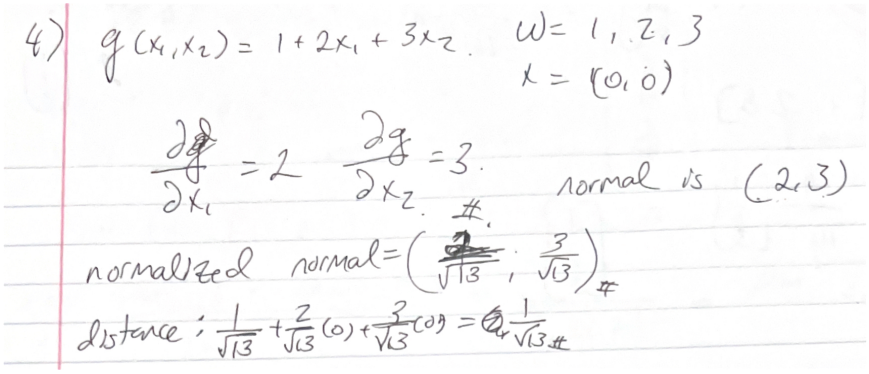
1.



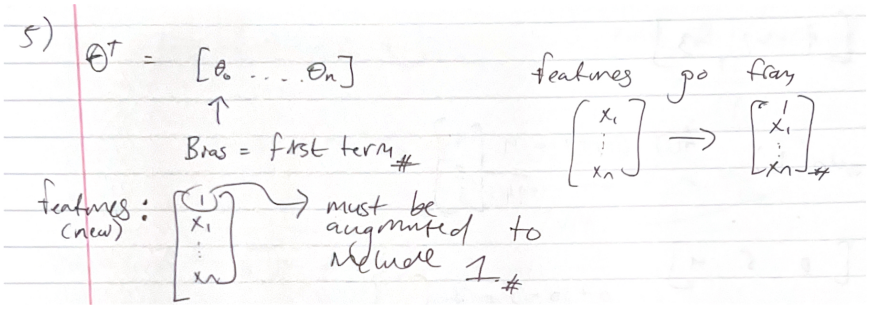
2. For a linear discriminant, on one side the value is greater than 0, the other side has values less than 0 and on the boundary itself the value is 0.

3. θ0 is the bias term and the distance from the decision boundary to the origin, whereas θ1 and θ2 are the normal vectors for the decision boundary.

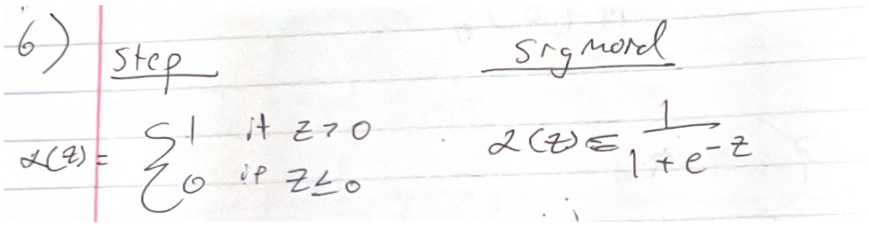
4.



5.



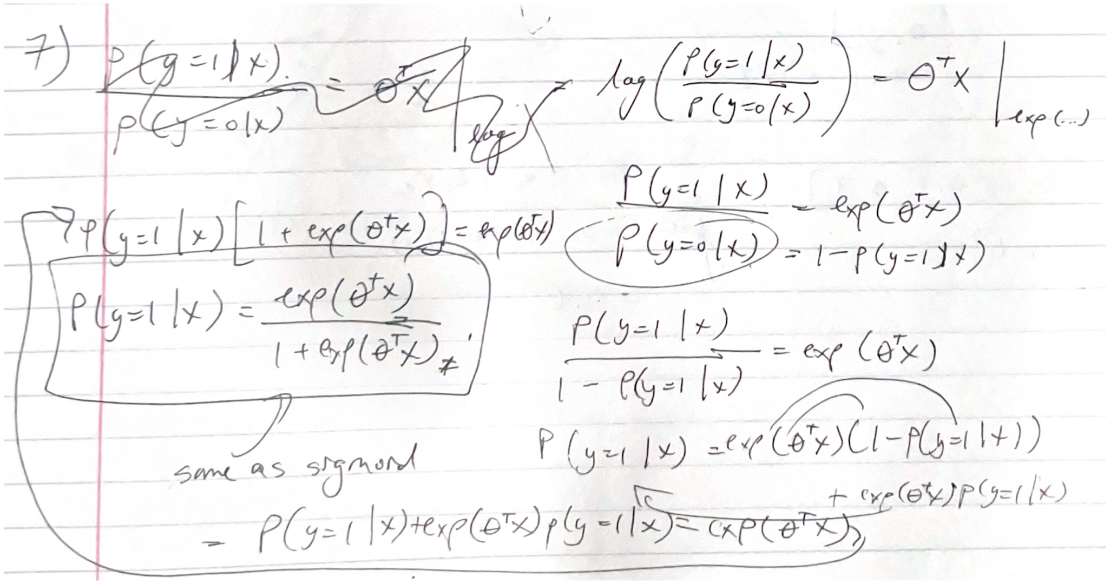
6.



The advantage of using sigmoid activation over the step function is that the sigmoid function you can take derivatives for easily and attain useful information but for the step function the

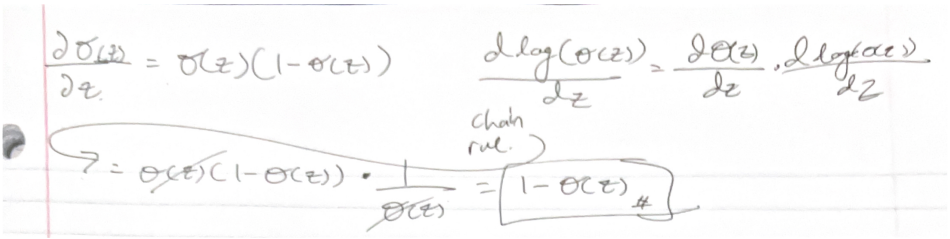
derivative is going to be 0. The sigmoid also can be used to output probabilities so that can come in handy when we deal with classification problems. The step function is way more decisive as-well which can be a bad thing if there's something like 0.00001, it'll instantly assign it to a 1 even though it's significantly closer to 0.
When θ is too small then the slope is a lot more flatter and less decisive (difficult to decide class).

7.



The linear discriminant θTx is used to model the log-likelihood ratio and by rearranging and mathematical manipulation you can obtain the sigmoid function equation in the box.

8.



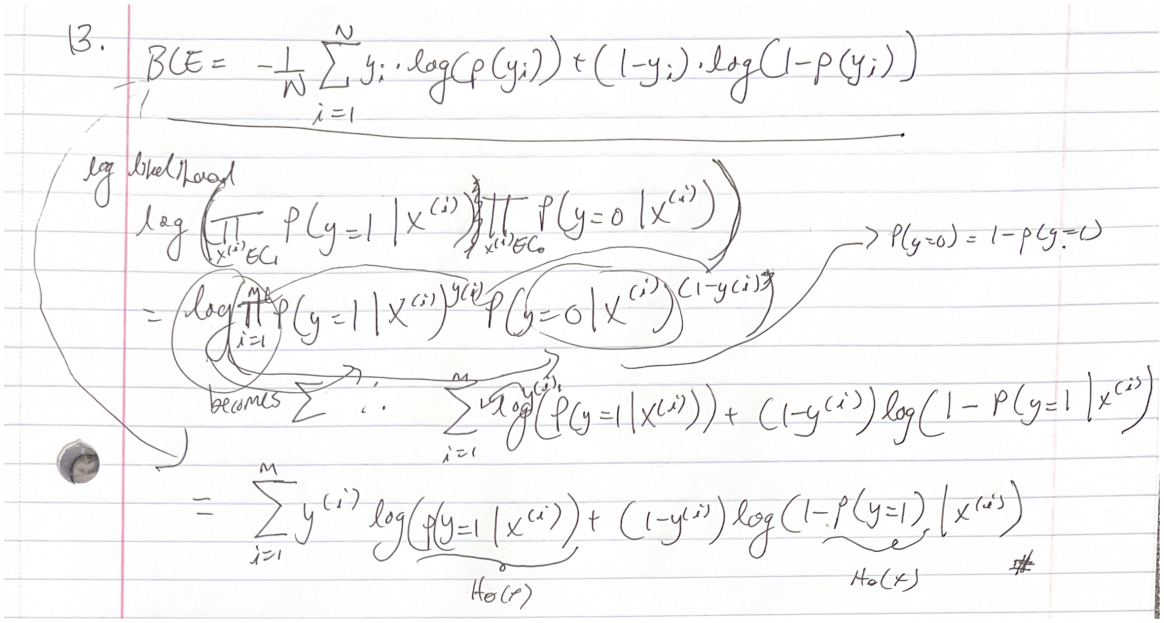
9. The direction is computed by calculating the gradient, the derivative of the loss function. The size of the update is controlled by the learning rate.

10. You can stop when you're at the global minimum of the function (it could also get stuck at local minima however). The condition could be numerically tracked by the difference between the current gradient and the previous gradient and see if it's less than a certain threshold (i.e. the change is very small). This will ensure that there isn't any significant change in the gradient and that it may have indeed reached a local or global minima.

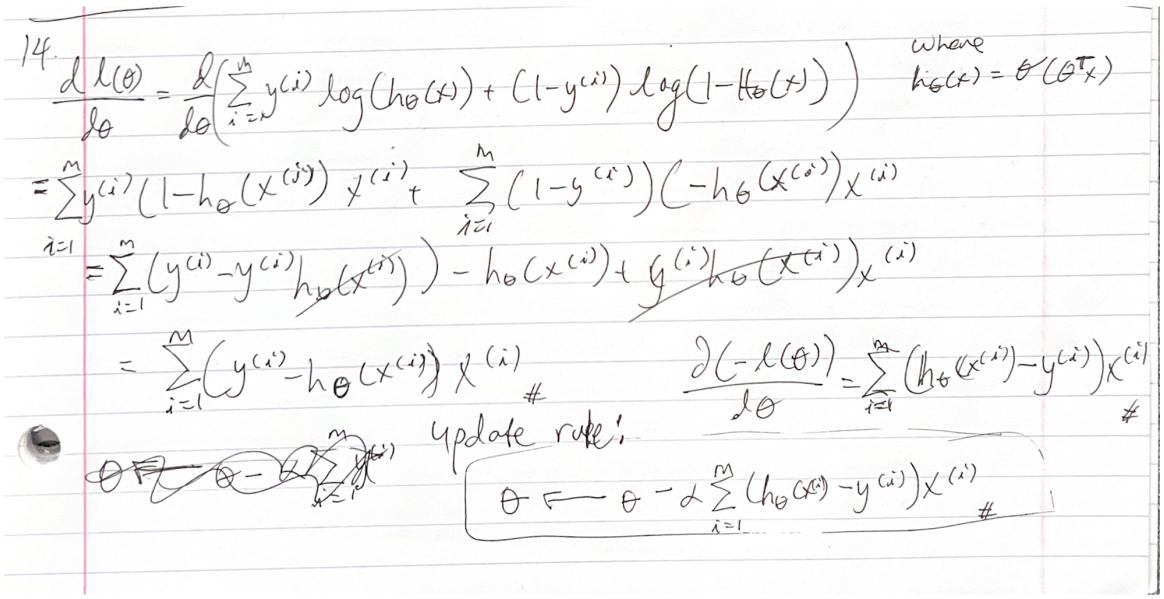
11. If a learning rate is too large then the algorithm will diverge, if the learning rate is too small it will take too small of steps and a much longer time to converge (may not ever converge) and plus it can get stuck in a local minima.

12. The empirical error loss calculates the total number of errors, it's not differentiable as it is a piecewise function and hence it can't be used in gradient descent.

13.



14.

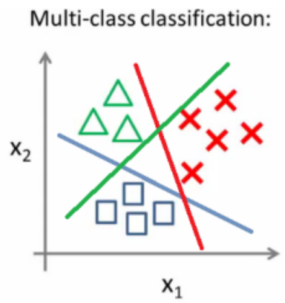


The meaning of the update equation is that it will advance the algorithm in search of the minima in the direction of the gradient which was computed by taking the derivative of the BCE loss function.

15.

<u>One against all</u>

- One against all tries to separate each individual class by having a discriminate function for each respective class.
- Therefore, there are 'k' discriminant functions where k is the number of classes
- You can imagine this by there being three clusters of data spaced out and there are exactly three lines that run and separate the three clusters of data, those three lines are the three respective discriminate functions:



<u>One against each other</u>

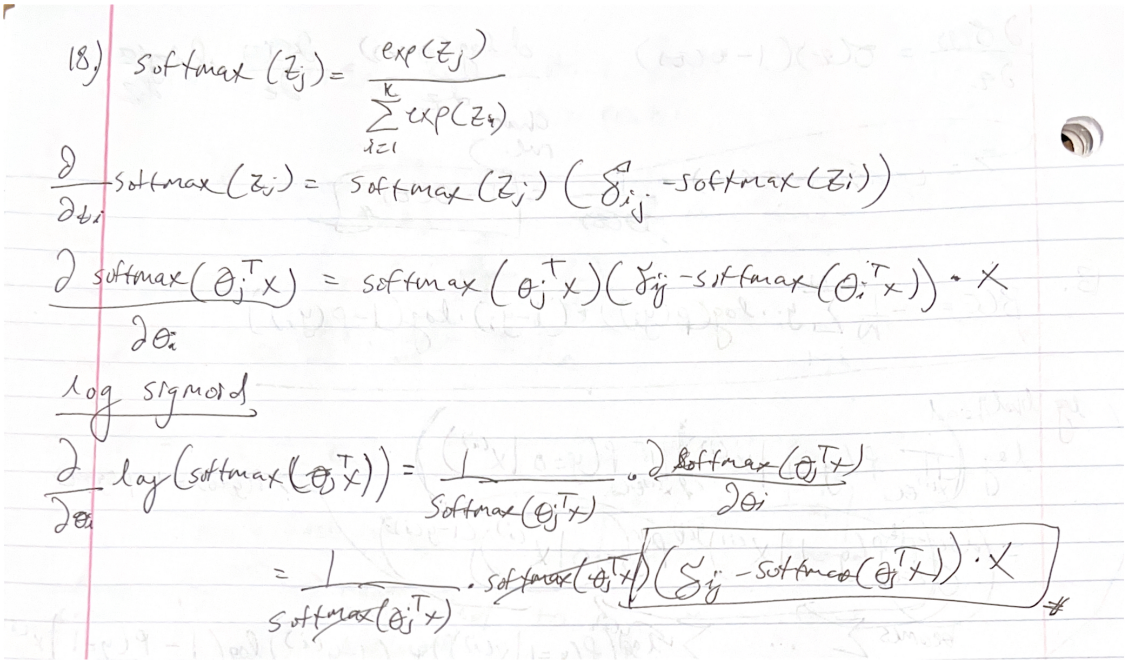
- One against each other aims to tackle one class at a time and derive a discriminate function to ensure that that singular class is best optimized and separated and that the discriminate function achieves high performance when separating the class
- There are a total of k(k-1)/2 total models / discriminate functions that we must examine where k is the number of classes
- The primary dataset gets split into one binary classification set for each pair of classes.

One against each other is easier to discriminate the data.

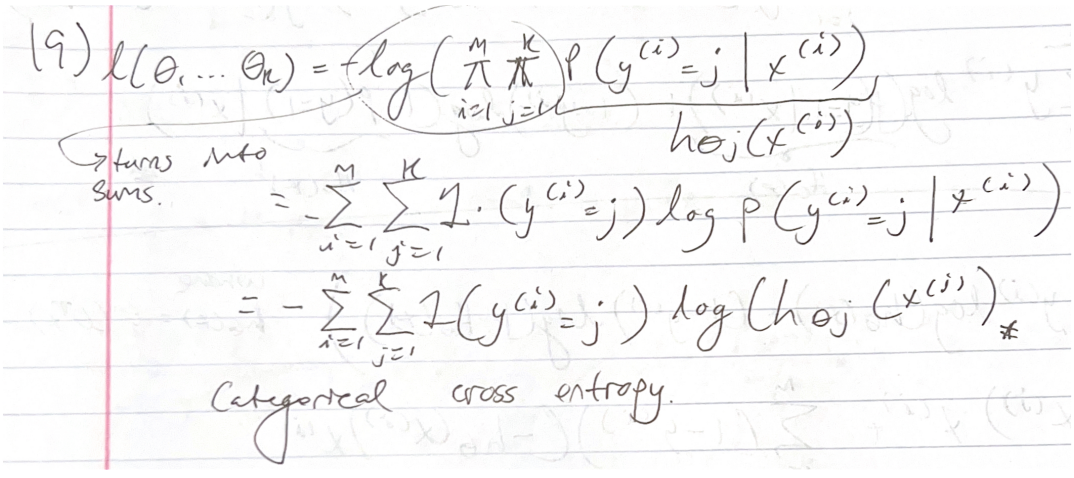
16. The rows of θT are the templates and there is effectively one template per class. θTx measures how similar the features match to each of the template and concurrently, high similarity can be indicative of high membership in the respective class. Multiple linear discriminant functions means you have multiple templates per each class.

17. Using softmax is an effective way to represent probabilities, all the class outputs will be in the interval 0,1 and the entirety of the class outputs would add up to 1. The larger the value in the range of 0,1 the higher the probability and certainty of it being that class

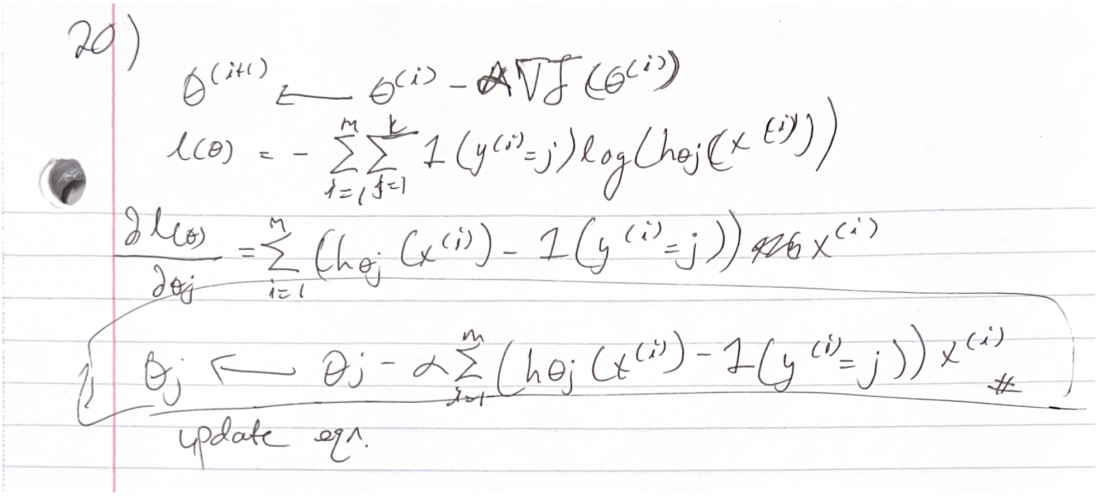
18.



19.



20.



The update rule shows how the new value of the parameter theta is going to be the old value plus an added factor adjusting it based on the partial derivative of the loss function for categorical cross entropy which in other words is also known as the gradient. The update equation is iteratively adjusted until it reaches either a local minima or a global minima. The parameter 'alpha' is the learning rate which is responsible for the size of the adjustment and is a hyper-parameter that gets tuned. The update equation shows that the softmax will output multiple classes as it looks at the 'k' classes.

Neural Networks

1. It maps the inputs to a higher dimensional space in an effort to find finer relationships between the original inputs. The benefit of this process is that it is easier to find relationships between features when expanding the dimension of said features and analyzing between them in the higher dimensional space.

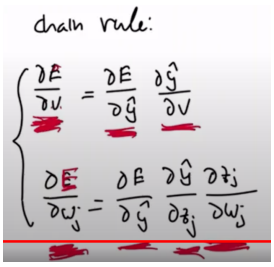
2. Dimensionality reduction, it compresses the input features into a lower dimensional space. The purpose and benefit of this process can be to extract solely the useful features or extract the features that have useful information encoded within them.

3. The gradients will differ between the two layers. The values z and Z_hat are unknown and so we can't get the expected values we want from the hidden layer unless there are additional steps that we follow up with. You can see from the following picture that the chain rule for the two layers differs:

$$v_j \leftarrow v_j - \eta\sum_{i=1}^{m}(\hat{y}_j^{(i)} - 1(y^{(i)}=j))z^{(i)}$$

$$w_j \leftarrow w_j - \eta\sum_{i=1}^{m}(\hat{z}_j^{(i)} - 1(z^{(i)}=j))x^{(i)}$$

$\hat{z}_j^{(i)} = ?$ $\quad z^{(i)} = ?$



4.

4) MSE $E = \frac{1}{2}\sum_{i=1}^{k}(y^{(i)} - \hat{y}^{(i)})^2$ $\hat{y} = z$ $z = \sigma(w_j^T x)$

hidden layer: $\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w_j}$

$$-\sum_{i=1}^{k}(y^{(i)} - \hat{y}^{(i)}) \cdot 1 \cdot \sigma(w_j^T x)(1 - \sigma(w_j^T x)) \cdot x^{(i)} \quad \#$$

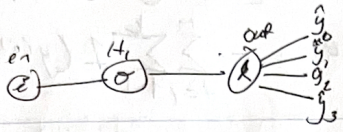
output layer:

$$\frac{\partial E}{\partial V} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V} = -\sum_{i=1}^{k}(y^{(i)} - \hat{y}^{(i)}) \cdot 1 \quad \#$$

5.

(5) outputs $\hat{y}_j = V_j^T z = V_{j0} + V_{j1}z_1 + V_{j2}z_2 + \dots V_{jn}z_n$

$\hookrightarrow z_j = \text{sigmoid}(w_j^T x)$



loss, $E = \frac{1}{2}\sum_{i=1}^{M}\sum_{e=1}^{K}(\hat{y}_e^{(i)} - y_e^{(i)})^2$

output

$$\frac{\partial E}{\partial V_j} = \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial V_j}$$

$$\frac{\partial E}{\partial \hat{y}_j} = 2 \cdot \frac{1}{2}\sum_{i=1}^{M}(\hat{y}_j^{(i)} - y_j^{(i)})^2 \quad \#$$

$$\frac{\partial \hat{y}_j}{\partial V_j} = z^{(i)}$$

$$= \sum_{i=1}^{M}(\hat{y}_j^{(i)} - y_j^{(i)})^2 \cdot z^{(i)} \quad \#$$

Hidden layer

$$\frac{\partial E}{\partial w_j} = \sum_{q=1}^{k}\frac{\partial E}{\partial \hat{y}_q} \cdot \frac{\partial \hat{y}_q}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_j}$$

$$\frac{\partial E}{\partial \hat{y}_q} = \sum_{i=1}^{m}(\hat{y}_q^{(i)} - y_q^{(i)})^2 \qquad \frac{\partial \hat{y}_q}{\partial z_j} = V_{jq}$$

$$\frac{\partial z_j}{\partial w_j} = z_j^{(i)}(1 - z_j^{(i)})x^{(i)}$$

$$\cdot \cdot \sum_{q=1}^{k}\sum_{i=1}^{\hat{}}(\hat{y}_q^{(i)} - y_q^{(i)})^2 \cdot V_{jq} \cdot z_j^{(i)}(1 - z_j^{(i)})x^{(i)}$$

6.

6) loss : Binary cross entropy → $E = \sum_{j=1}^{k} y_j^{(i)} \log(\hat{y}_j^{(i)})$

hidden layer : sigmoid $z = \sigma(w^T x)$

output : sigmoid. $\hat{y} = \sigma(v^T z)$

$\partial \hat{y} = \hat{y}(1-\hat{y})$

hidden : $\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_j}$

$\frac{\partial E}{\partial \hat{y}} = f'g + fg' = \hat{y}_j^{(i)}(1-\hat{y}_j^{(i)}) \log(\hat{y}_j^{(i)}) + y_j^{(i)} \cdot \frac{1}{\hat{y}_j^{(i)}} \hat{y}_j^{(i)}(1-\hat{y}_j^{(i)})$

$= \hat{y}_j^{(i)}(1-y_j^{(i)}) \log(\hat{y}_j^{(i)}) + y_j^{(i)}(1-\hat{y}_j^{(i)})$

$\frac{\partial \hat{y}}{\partial z_j} = \sigma(v^T z)(1-\sigma(v^T z)) \cdot v_j = \hat{y}(1-\hat{y}) \cdot v_j$

$\frac{\partial z_j}{\partial w_j} = z_j(1-z_j) \cdot x^{(i)}$

$\sum_{i=1}^{m} \hat{y}_j^{(i)}(1-y_j^{(i)}) \log(\hat{y}_j^{(i)}) + y_j^{(i)}(1-\hat{y}^{(i)}) \cdot \hat{y}(1-\hat{y}) \cdot v_j \cdot z_j^{(i)}(1-z_j^{(i)}) \cdot x^{(i)}$

output layer:

$\frac{\partial E}{\partial v_j} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial v} = \sum_{i=1}^{m} \hat{y}_j^{(i)}(1-y_j^{(i)}) \log(\hat{y}_j^{(i)}) + y_j^{(i)}(1-\hat{y}) \cdot \hat{y}(1-\hat{y}) \cdot z_j^{(i)}$ #

7.

7) multi-class

hidden : sigmoid

output : softmax.

L = cat. cross entr → true label

$L = \sum_i (y_i) \log(\hat{y}_i)$ → probability, it's for softmax output.

$\frac{\partial E}{\partial w_j} = \sum_{i=1}^{k} \frac{\partial E}{\partial \hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_j} = \sum_{q=1}^{m} \sum_{i=1}^{k} (\hat{y}_i^{(q)} - y^{(q)}) v_{iq} z_j^{(i)}(1-z_j^{(i)}) x^{(i)}$

output layer $= \frac{\partial E}{\partial v_j} = \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial v_j} = \sum_{i=1}^{m} (\hat{y}_j^{(i)} - y_j^{(i)}) z^{(i)}$ #

8. The weights are initialized randomly but they must not be large values, they need to be closer to 0. The aim of the initialization is to provide a good starting point for the optimization algorithm to take over and ensure that the weights either don't explode or vanish within the first few iterations. If the weights are initialized all as 0 then the neurons will learn the same features during training, in-fact any initialization scheme that is initializing the weights to be constant will all result in the same weights being trained. You need the weights to be closer to 0 so they don’t vanish and/or explode and different so that the weights all learn and pick-up different features and values.

Computation Graphs

1. The advantage of using computation graphs is that it allows for easier computation of the gradients in order to update the weights and optimize them for the network. Computation graphs provide a way to optimize the computation and so that it is also more intuitive for someone to look at and quickly and easily be able to figure out the gradients as needed for back-prop.

In the forward pass the weights and features are passed through the network and the activation functions are applied to the weights and features as designated to each node. Whereas in the backward pass the gradients are calculated with respect to the nodes and each input.

Each node in the graph needs to be able to compute a value (an activation) based on the features / data it's being fed in and this value is configured by the user (so the user can choose softmax, sigmoid, linear, relu, whatever activation they want). In addition, in the backward pass the each node must be able to compute the gradients, which is the derivatives for a specific path (the derivative with respect to a feature / input).

Each node needs to store the outputs and the values from the forward pass so that it can compute the gradients based off that and adjust the values as calculated from the gradient to optimize the weights for the network.

2.

2) hidden out: $z = f_1(w_1, x)$ $\quad \hat{y} = f_2(w_2, f_1(w_1, x))$ #

output out: $\hat{y} = f_2(w_2, z)$ $\quad E = \frac{1}{2}\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)})^2$

$$E = \frac{1}{2}\sum_{i=1}^{M}\left(y^{(i)} - f_2(w_2, f_1(w_1, x))\right)^2 \quad \#$$

$\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}$ $\quad \frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w_1} = \frac{\partial E}{\partial f_2} \cdot \frac{\partial f_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial w_1}$ not specific

$$\frac{\partial E}{\partial \hat{y}} = \frac{1}{2}\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)})(-1) = -\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)}) \quad \#$$

$\frac{\partial \hat{y}}{\partial z} \to \partial(f_2)$ $\quad \frac{\partial \hat{y}}{\partial z} = \frac{\partial f_2}{\partial f_1}$ # $\quad \frac{\partial z}{\partial w_1} = \frac{\partial f_1}{\partial w_1}$

L2 Loss

since $z = f_1(\dots)$, $\partial z = \partial(f_1)$ ... same logic applies for $\partial \hat{y}$.

$$\frac{\partial E}{\partial w_1} = -\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)}) \cdot \frac{\partial f_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial w_1} = -\sum_{i=1}^{M}\left(y^{(i)} - f_2(w_2, f_1(w_1, x))\right) \cdot \frac{\partial f_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial w_1} \quad *$$

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w_2} = -\sum_{i=1}^{M}(y^{(i)} - \hat{y}^{(i)}) \cdot \frac{\partial f_2}{\partial w_2} \quad \# = -\sum_{i=1}^{M}(y^{(i)} - f_2(w_2, z)) \cdot \frac{\partial f_2}{\partial w_2} \quad \#$$

---

2. cross entropy $L = -\sum y_i \ln(\hat{y}_i)$ $\quad \hat{y} = f_2(w_2, f_1(w_1, x))$

$$L = \sum_{i=1}^{M} y^{(i)} \log\left(f_2(w_2, f_1(w_1, x^{(i)}))\right) + (1 - y^{(i)}) \log\left(1 - f_2(w_2, f_1(w_1, x^{(i)}))\right)$$

hidden out $= f_1(w_1, x) = z$, out out $= \hat{y}$.

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial f_2} \cdot \frac{\partial f_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial w_1} \quad \#$$

$$\frac{\partial L}{\partial f_2} = y^{(i)} \cdot \frac{1}{f_2(w_2, f_1(w_1, x^{(i)}))} + (-1) \cdot (1 - y^{(i)}) \cdot \frac{1}{1 - f_2(w_2, f_1(w_1, x^{(i)}))}$$

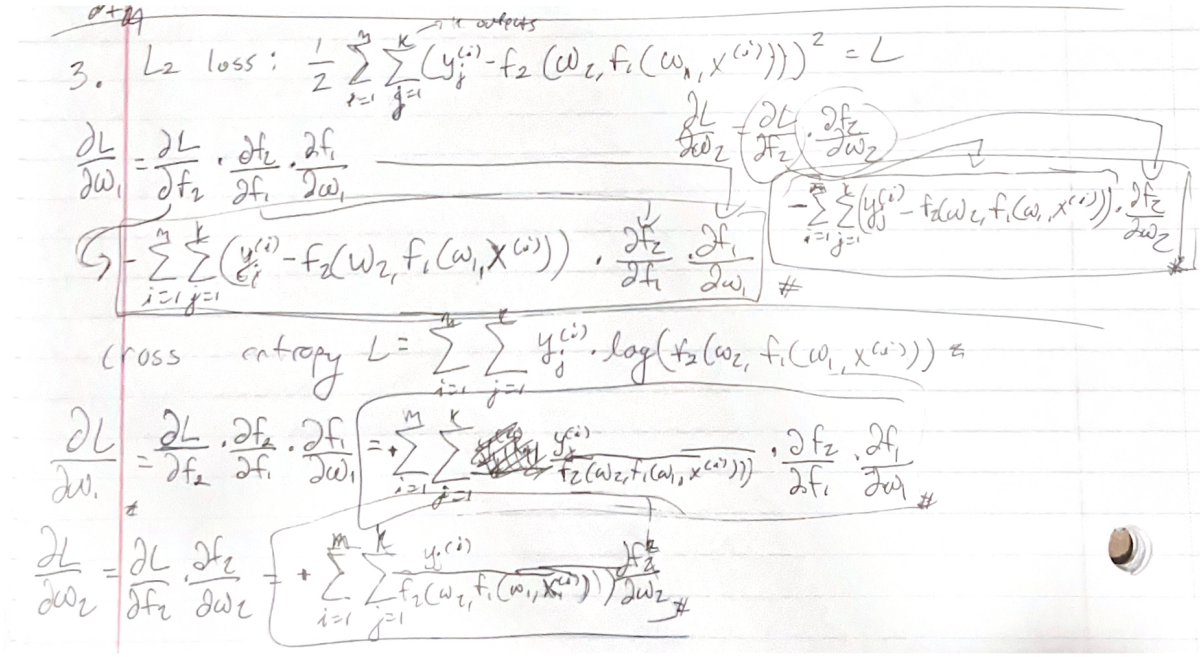
$$= \sum_{i=1}^{M} \frac{y^{(i)}}{f_2(w_2, f_1(w_1, x^{(i)}))} + \frac{1 - y^{(i)}}{1 - f_2(w_2, f_1(w_1, x^{(i)}))} \cdot \frac{\partial f_2}{\partial f_1} \cdot \frac{\partial f_1}{\partial w_1} \quad *$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial f_2} \cdot \frac{\partial f_2}{\partial w_2}$$

$$= \sum_{i=1}^{M} \frac{y^{(i)}}{f_2(w_2, f_1(w_1, x^{(i)}))} - \frac{1 - y^{(i)}}{1 - f_2(w_2, f_1(w_1, x^{(i)}))} \cdot \frac{\partial f_2}{\partial w_2} \quad \#$$

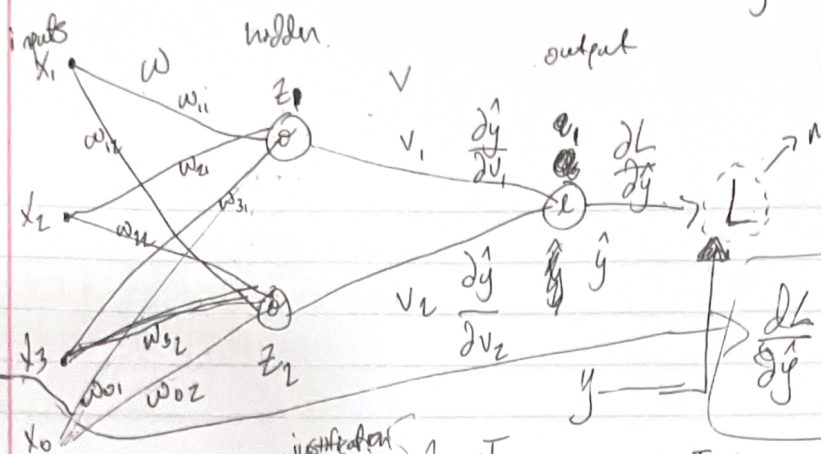
$\frac{\partial L}{\partial f_2}$

3.


3. L2 loss: $\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{k}(y_j^{(i)} - f_2(w_2, f_1(w_1, x^{(i)})))^2 = L$
k outputs
$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial f_2}\cdot\frac{\partial f_2}{\partial f_1}\cdot\frac{\partial f_1}{\partial w_1}$
$\frac{\partial L}{\partial w_2}$ $\left(\frac{\partial L}{\partial f_2}\cdot\frac{\partial f_2}{\partial w_2}\right)$
$-\sum_{i=1}^{m}\sum_{j=1}^{k}(y_j^{(i)} - f_2(w_2, f_1(w_1, x^{(i)})))\cdot\frac{\partial f_2}{\partial w_2}$
$-\sum_{i=1}^{m}\sum_{j=1}^{k}(y_j^{(i)} - f_2(w_2, f_1(w_1, x^{(i)}))\cdot\frac{\partial f_2}{\partial f_1}\cdot\frac{\partial f_1}{\partial w_1}$
cross entropy $L = \sum_{i=1}^{m}\sum_{j=1}^{k} y_j^{(i)}\cdot\log(f_2(w_2, f_1(w_1, x^{(i)})))$
$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial f_2}\cdot\frac{\partial f_2}{\partial f_1}\cdot\frac{\partial f_1}{\partial w_1} = +\sum_{i=1}^{m}\sum_{j=1}^{k}\frac{y_j^{(i)}}{f_2(w_2, f_1(w_1, x^{(i)}))}\cdot\frac{\partial f_2}{\partial f_1}\cdot\frac{\partial f_1}{\partial w_1}$
$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial f_2}\cdot\frac{\partial f_2}{\partial w_2} = +\sum_{i=1}^{m}\sum_{j=1}^{k}\frac{y_j^{(i)}}{f_2(w_2, f_1(w_1, x^{(i)}))}\cdot\frac{\partial f_2}{\partial w_2}$

4.

regression network



MSE or L_2 loss

$\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)})^2$

$\frac{\partial L}{\partial \hat{y}} = 2\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)})$ #

justification: $\hat{y} = V^T z$ $\quad z_j = \theta(w_j^T x)$

$\frac{\partial (V_0 + V_1 z_1 + \dots V_n z_n)}{V_1} = z_1$ *

gradients

V: $\frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V_1} = \frac{\partial L}{\partial V_1}$ # $= 2\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)}) \cdot z_1^{(i)}$

$\frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V_2} = \frac{\partial L}{\partial V_2}$ # $= 2\sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)}) \cdot z_2^{(i)}$ #

Bias: $\frac{\partial L}{\partial V_0} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V_0}$ # $= \frac{\partial L}{\partial \hat{y}} \cdot z_0^{(i)}$ *

w: $\frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_1} \cdot \frac{\partial z_1}{\partial w_{11}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial (V_0 + V_1 z_1 + V_2 z_2 + \dots V_n z_n)}{\partial z_1} = 0 + V_1 + 0 + 0 + \dots 0 = V_1$

proof/just.

$\frac{\partial z_1}{\partial w_{11}} = \frac{\partial (\theta(w_1^T x))}{\partial w_{11}} = \theta(w_{10} + w_{11} x_1 + w_{12} x_2 + w_{31} x_3)$ (derivative) $\to 0 + x_1 + 0 + 0 + \dots 0$

$= \theta(w_1^T x)(1 - \theta(w_1^T x)) \cdot x_1^{(i)}$

$z_1^{(i)}$ $\quad 1 - z_1^{(i)}$

$\frac{\partial L}{\partial w_{11}} = \frac{\partial L}{\partial \hat{y}} \cdot V_1 \cdot z_1^{(i)}(1 - z_1^{(i)}) \cdot x_1^{(i)}$ *

$\frac{\partial L}{\partial w_{12}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_2} \cdot \frac{\partial z_2}{\partial w_{12}} = \frac{\partial L}{\partial \hat{y}} \cdot V_2 \cdot z_2^{(i)}(1 - z_2^{(i)}) \cdot x_1^{(i)}$ #

$\frac{\partial L}{\partial w_{21}} = \frac{\partial L}{\partial \hat{y}} \cdot V_1 \cdot z_1^{(i)}(1 - z_1^{(i)}) \cdot x_2^{(i)}$ #

$\frac{\partial L}{\partial w_{31}} = \frac{\partial L}{\partial \hat{y}} \cdot V_1 \cdot z_1^{(i)}(1 - z_1^{(i)}) \cdot x_3^{(i)}$ #

$\frac{\partial L}{\partial w_{32}} = \frac{\partial L}{\partial \hat{y}} \cdot V_2 \cdot z_2^{(i)}(1 - z_2^{(i)}) \cdot x_3^{(i)}$ #

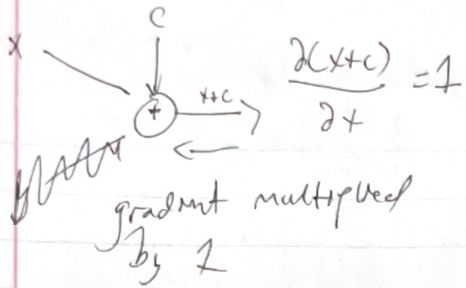
$\frac{\partial L}{\partial w_{22}} = \frac{\partial L}{\partial \hat{y}} \cdot V_2 \cdot z_2^{(i)}(1 - z_2^{(i)}) \cdot x_2^{(i)}$ #

Bias

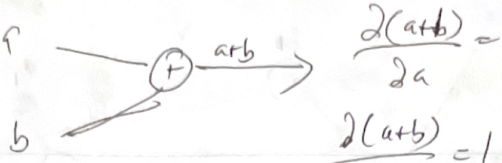
$\frac{\partial L}{\partial w_{01}} = \frac{\partial L}{\partial \hat{y}} \cdot V_1 \cdot z_1^{(i)}(1 - z_1^{(i)}) \cdot 1$

$\frac{\partial L}{\partial w_{02}} = \frac{\partial L}{\partial \hat{y}} \cdot V_2 \cdot z_2^{(i)}(1 - z_2^{(i)}) \cdot 1$ #

5, addition of a constant

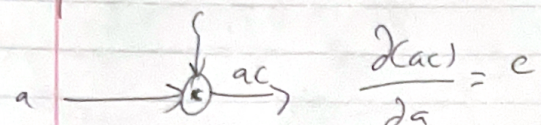


addition of two inputs



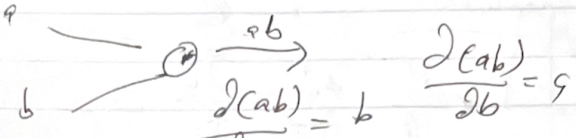
In both cases, gradient is multiplied by 1.

multiplication of a constant



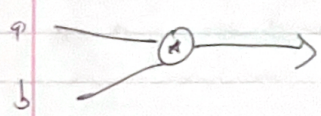
gradient multiplied by constant 'c'

Mult. of two inputs



gradient multiplied by the other input.

max of two inputs



$$\max(a,b) = \begin{cases} a & \text{if } a \geq b \\ b & \text{otherwise} \end{cases}$$

$$\frac{\partial(\max(a,b))}{\partial a} = \begin{cases} 1 & \text{if } a \geq b \\ 0 & \text{else} \end{cases}$$

$$\frac{\partial(\max(a,b))}{\partial b} = \begin{cases} 0 & \text{if } a \geq b \\ 1 & \text{else} \end{cases}$$

the gradient is multiplied by 1 for the maximum input only.

min of two inputs

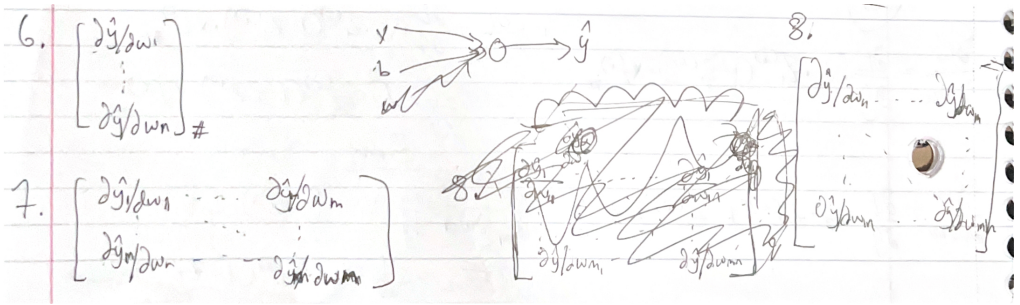
$$\min(a,b) = \begin{cases} a & \text{if } a \leq b \\ b & \text{else} \end{cases}$$

for min of two inputs, the gradient will also be multiplied by 1 for the minimum input only

6. If the input node is a vector (rank 1) and the output is a scalar (rank 0) then the derivative that will flow back will be a tensor of rank (1) + (0) = 1.

7. If the input and output nodes are both vectors (both rank 1) then the rank of the gradient tensor will be (1) + (1) = rank 2 tensor flowing back.

8. If the input node is a matrix (rank 2) and the output is a scalar (rank 0) then the rank of the gradient tensor will be 2+0 = 2.



9. $(F \circ G) = F(g(x))$ $\frac{\partial (F \circ G)}{\partial g} = \frac{\partial F}{\partial g} \cdot \frac{\partial g}{\partial}$

let $F = F(x_0, x_1, \dots x_n) = \begin{bmatrix} f_1(x_0, x_1, \dots x_n) \\ \vdots \\ f_m(x_0, x_1, \dots x_n) \end{bmatrix}$

let $g = G(x_0, x_1, \dots x_n) = \begin{bmatrix} g_1(x_0, x_1, \dots x_n) \\ \vdots \\ g_m(x_0, x_1, \dots x_n) \end{bmatrix}$

and $\therefore (F \circ G) = F(G) = \begin{bmatrix} f_1(g_1(x_0) \dots g_1(x_n)) \\ \vdots \\ f_m(g_m(x_0) \dots g_m(x_n)) \end{bmatrix}$

$\nabla F \circ G|_x = \text{Jacobian}$

$= \begin{bmatrix} \nabla f_1(g_1(x_0) \dots g_1(x_n)) \\ \vdots \\ \nabla f_m(g_m(x_0), \dots g_m(x_n)) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1(g_1(x))}{x_0} & \dots & \frac{\partial f_1(g_1(x))}{x_n} \\ \vdots & & \\ \frac{\partial f_m(g_m(x))}{x_0} & \dots & \frac{\partial f_m(g_m(x))}{x_n} \end{bmatrix}$

10.

10. $F(x,y,z) = [3xy \quad y-z]^T$ $(F \circ G)(x,y) = F(g(x,y))$

$G(x,y) = [x-5y \quad xy \quad x-y]^T$ (x, y, z)

Jacobian: $\begin{bmatrix} 6xy-15y^2 & 3x^2-30xy \\ y-1 & x+1 \end{bmatrix}_{\#}$

$\begin{bmatrix} 3(x-5y)(xy) & xy-(x-y) \end{bmatrix}$

$(3x-15y)(xy) \to f_1$ $\to f_2$ directly

$(F \circ G)(x,y) = \begin{bmatrix} 3x^2y-15xy^2 & xy-x+y \end{bmatrix}^T_{\#} \to \nabla F \circ G(x,y) = \frac{\partial f_1}{\partial x} = 6xy-15y^2 \quad \frac{\partial f_1}{\partial y} = 3x^2-30y$

$\partial f_2/\partial x = y-1 \quad \frac{\partial f_2}{\partial y} = x+1$

10. chain rule $f(x,y,z) = \begin{bmatrix} 3xy \\ y-z \end{bmatrix}$ $G(x,y) = \begin{bmatrix} x-5y \\ xy \\ x-y \end{bmatrix}_{\#} \to 3\times 1$

$J = \text{Jacobian}$

$J_{f \circ g}(x,y) = J_f(g(x,y)) \cdot J_g(x,y) \Leftrightarrow$ chain rule

first lets compute $J_G(x,y) = \begin{bmatrix} 1 & -5 \\ y & x \\ 1 & -1 \end{bmatrix}$ dim = 3x2

now compute $J_f(g(x,y))$ $\to = J_f(x,y) \circ G(x,y) = \begin{bmatrix} 3y & 3x & 0 \\ 0 & 1 & -1 \end{bmatrix}$ dim = 2x3

$J_f(x,y,z) \circ G(x,y) = \begin{bmatrix} 3xy & 3x-15y & 0 \\ 0 & 1 & -1 \end{bmatrix}$ dim = 2x3 $= J_F(g(x,y))$ #

finally: $J_{f \circ g}(x,y) = J_F(g_1(x,y,z)) \cdot J_G(x,y) = \begin{bmatrix} 3xy & 3x-15y & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 & -5 \\ y & x \\ 1 & -1 \end{bmatrix}$ $\to$

10 continued.

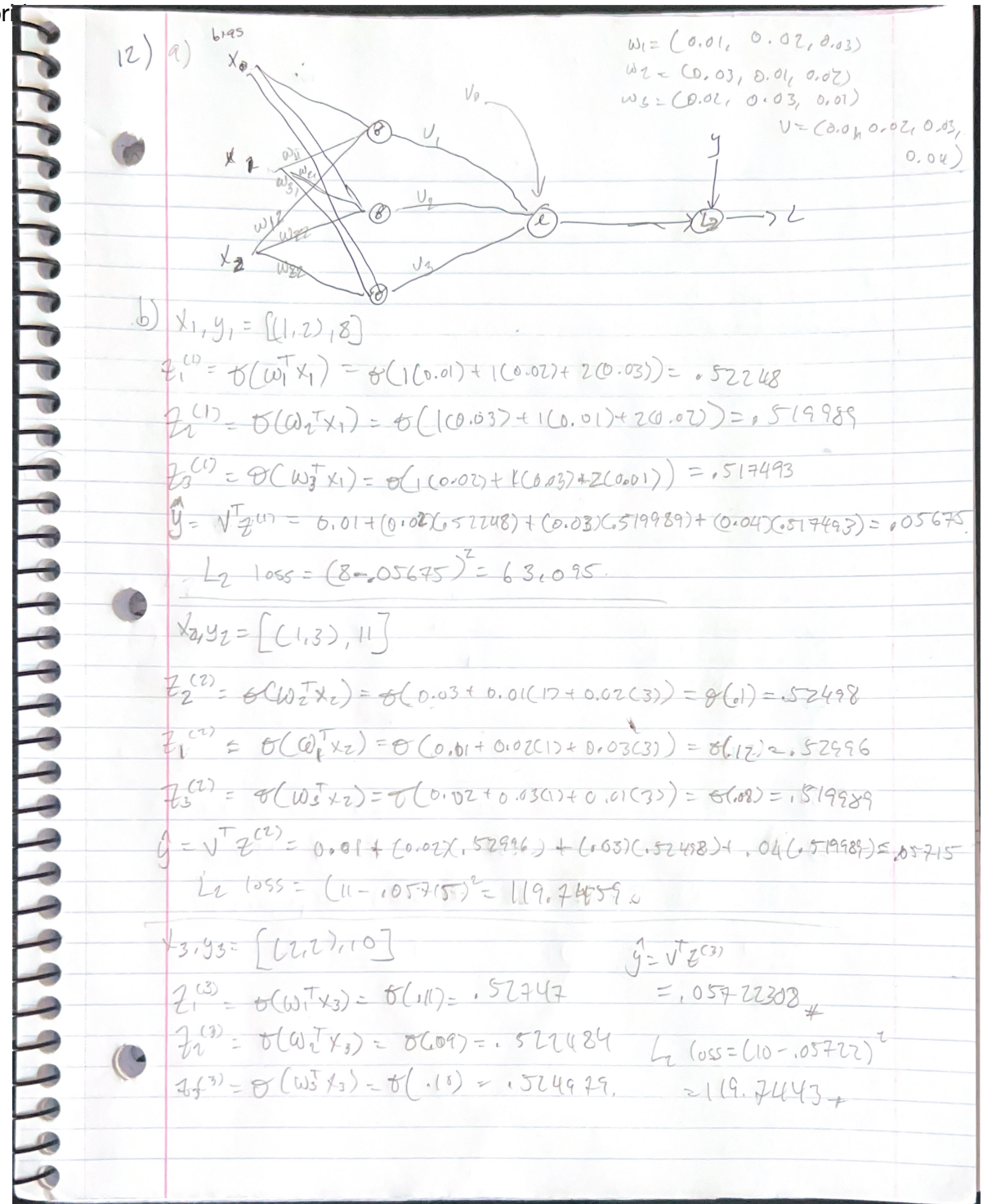
$$\rightarrow \begin{bmatrix} 3xy & 3x-15y & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 & -5 \\ y & x \\ 1 & -1 \end{bmatrix} =$$

$$\begin{bmatrix} 3xy + 3xy - 15y^2 + 0 & -15xy + 3x^2 - 15xy + 0 \\ 0 + y - 1 & 0 + x + 1 \end{bmatrix} = \begin{bmatrix} 6xy - 15y^2 & 3x^2 - 30y \\ y - 1 & x + 1 \end{bmatrix} \#$$

Same as other calculated via direct method ✓

11. It is important to order the nodes in order to properly take the necessary derivatives to compute the gradient for the update rule so that we can optimally converge the optimization algori

12.



12 continued.

12c) $\frac{\partial L_2}{\partial(\omega_1,\omega_2,\omega_3)} = \frac{\partial L_2}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial v} \cdot \frac{\partial v}{\partial(\omega_1,\omega_2,\omega_3)}$ find $\frac{\partial L_2}{\partial \omega_1}, \frac{\partial L_2}{\partial \omega_2}, \frac{\partial L_2}{\partial \omega_3}$

$\frac{\partial L_2}{\partial v} = \frac{\partial L_2}{\partial \hat{y}^{(i)}} \cdot \frac{\partial \hat{y}^{(i)}}{\partial v}$ $\frac{\partial L_2}{\partial v}$

$\frac{\partial L_2}{\partial v} = \sum_{i=1}^{m} \left[(\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}) \cdot v \right] - y^{(i)} \cdot (\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}))$

$\frac{\partial L_2}{\partial(\omega_1,\omega_2,\omega_3)} = \sum_{i=1}^{m} \left[(\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}) \cdot V \cdot \partial z^T / \partial x \right]$

$= \begin{bmatrix} \sigma(\omega_1 x^{(i)})(1-\sigma(\omega_1 x^{(i)})) \cdot x^{(i)} \\ \sigma(\omega_2 x^{(i)})(1-\sigma(\omega_2 x^{(i)})) \cdot x^{(i)} \\ \sigma(\omega_3 x^{(i)})(1-\sigma(\omega_3 x^{(i)})) \cdot x^{(i)} \end{bmatrix}$ #

12c) $\frac{\partial L_2}{\partial(\omega_1,\omega_2,\omega_3)} = \frac{\partial L_2}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial v} \cdot \frac{\partial v}{\partial(\omega_1,\omega_2,\omega_3)}$ find $\frac{\partial L_2}{\partial \omega_1}, \frac{\partial L_2}{\partial \omega_2}, \frac{\partial L_2}{\partial \omega_3}$

$\frac{\partial L_2}{\partial v} = \frac{\partial L_2}{\partial \hat{y}^{(i)}} \cdot \frac{\partial \hat{y}^{(i)}}{\partial v}$ $\frac{\partial L_2}{\partial v}$

$\frac{\partial L_2}{\partial v} = \sum_{i=1}^{m} \left[(\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}) \cdot v \right] - y^{(i)} \cdot (\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}))$

$\frac{\partial L_2}{\partial(\omega_1,\omega_2,\omega_3)} = \sum_{i=1}^{m} \left[(\sigma(\omega_1 x^{(i)}), \sigma(\omega_2 x^{(i)}), \sigma(\omega_3 x^{(i)}) \cdot V \cdot \partial z^T / \partial x \right]$

$= \begin{bmatrix} \sigma(\omega_1 x^{(i)})(1-\sigma(\omega_1 x^{(i)})) \cdot x^{(i)} \\ \sigma(\omega_2 x^{(i)})(1-\sigma(\omega_2 x^{(i)})) \cdot x^{(i)} \\ \sigma(\omega_3 x^{(i)})(1-\sigma(\omega_3 x^{(i)})) \cdot x^{(i)} \end{bmatrix}$ #

$\frac{\partial L_2}{\partial v_0} = (.05675)(z_0^{(i)}) = 7.94325 (\sigma(\omega)^T x_0)) =$

$\frac{\partial L_2}{\partial v_1} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) z_1^{(i)} = 7.94325 (.52248) + (11 - .05715)(.52996) + (10 - .057223)(.52744)$

≈ 15.19 #

$\frac{\partial L_2}{\partial v_2} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) z_2^{(i)} = 15.066$ $\frac{\partial L}{\partial v} = (15.19, 15.066, 15.017)$

$\frac{\partial L_2}{\partial v_3} = \vert\vert = 15.017$

$\frac{\partial L_2}{\partial \omega_1} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) \cdot v_1 \cdot z_1^{(i)} (1 - z_1^{(i)}) \cdot x^{(i)}$

$\frac{\partial L_2}{\partial \omega_{11}} = (2 - .05675) \cdot 0.02 \cdot (.52248)(1 - .52248) \cdot 1 + (11 - .05715) \cdot .02 \cdot (.52996) \cdot (1 - .52996) \cdot 1 + (11 - .057223) \cdot .02 \cdot .52743 \cdot (1 - .52743)(2)$

$.03964$ $.0545$ $.109097$

$= .19.33$ #

$\frac{\partial L_2}{\partial \omega_{12}} = .3422$ $\frac{\partial L_2}{\partial \omega_{10}} = .1439$ $\frac{\partial L_2}{\partial \omega_2} = (.216, .29, .5133)$ $\frac{\partial L}{\partial \omega_3} = \begin{bmatrix} .208 \\ .387 \\ .665 \end{bmatrix}$

13a) $f(x,y) = (2x+3y)^2$ $\nabla f(x,y)$

$= [2(2x+3y)\cdot 2,\ 2(2x+3y)\cdot 3]$

$= [8x+12y,\ 12x+18y]$

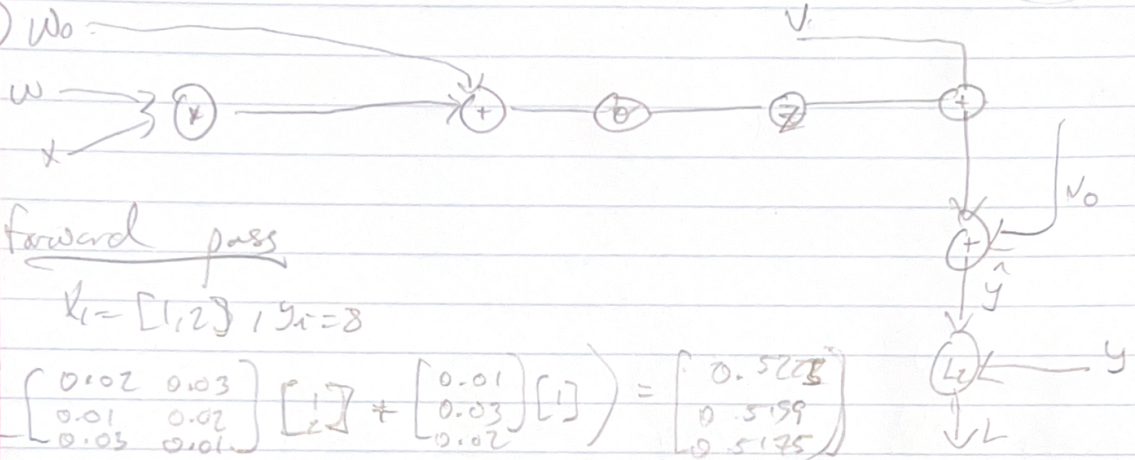
13b) $F(x,y) = \begin{bmatrix} x^2+2y \\ 3x+4y^2 \end{bmatrix}$ $J_F = \begin{bmatrix} 2x & 2 \\ 3 & 8y \end{bmatrix}\Big|_{(1,2)} = \begin{bmatrix} 2 & 2 \\ 3 & 16 \end{bmatrix}$ #

13c) FoG

$G(x) = \begin{bmatrix} x \\ x^2 \end{bmatrix}$ $F \circ G = F(g(x))$ $\begin{bmatrix} (x)^2 + 2(x^2) \\ 3(x) + 4(x^2)^2 \end{bmatrix} = \begin{bmatrix} x^2 + 2x^2 \\ 3x + 4x^4 \end{bmatrix} = \alpha$

$J_\alpha(x) = \begin{matrix} 2x+4x \\ 3+16x^3 \end{matrix} = \begin{bmatrix} 6x \\ 16x^3+3 \end{bmatrix}\Big|_{x=2} = \begin{bmatrix} 12 \\ 131 \end{bmatrix}$ #

13d) w_0 v_1



forward pass

$x_1 = [1,2]$, $y_1 = 8$

$\sigma\left(\begin{bmatrix} 0.02 & 0.03 \\ 0.01 & 0.02 \\ 0.03 & 0.01 \end{bmatrix}\begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 0.01 \\ 0.03 \\ 0.02 \end{bmatrix}[1]\right) = \begin{bmatrix} 0.5223 \\ 0.5199 \\ 0.5175 \end{bmatrix}$

$[0.02\ \ 0.03\ \ 0.04]\cdot 2] + 0.01$

$\hat{y} = 1.057$, $y = 8$ $L_2(\hat{y}, y) = 63.1$

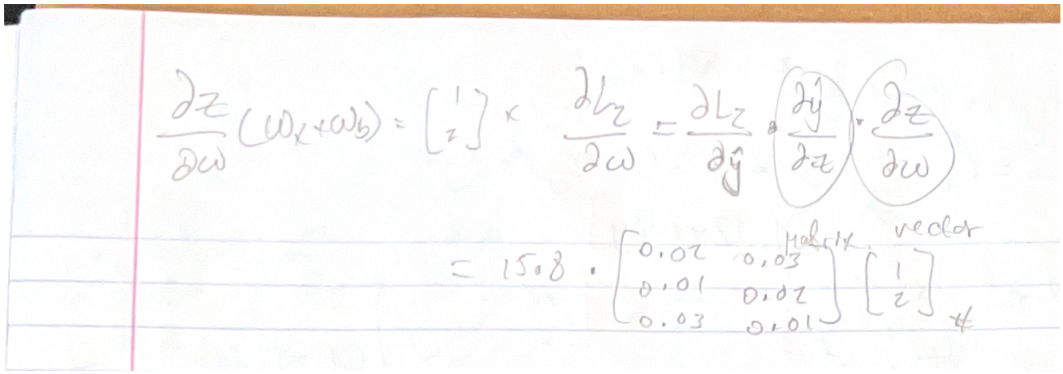
backward pass

$\frac{\partial L_2}{\partial \hat{y}} = 2(-8 - 0.057) = 15.8$, $\frac{\partial \hat{y}}{\partial V}(V_2 + V_b) = 2$.

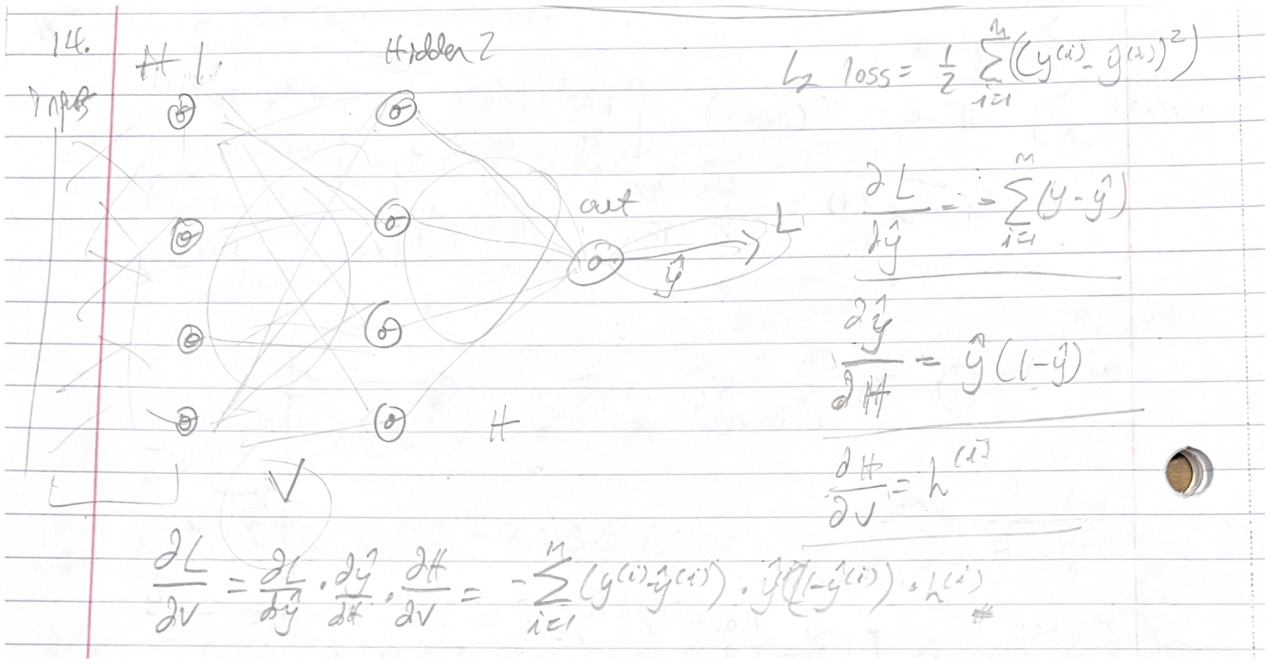
$\frac{\partial L_2}{\partial y} \cdot \frac{\partial \hat{y}}{\partial v} = 15.8 \begin{bmatrix} 0.5223 \\ 0.5195 \\ 0.5175 \end{bmatrix}$ $\frac{\partial \hat{y}}{\partial V_b} = 1$

$\frac{\partial L_2}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V_b} = +15.8$ $\frac{\partial \hat{y}}{\partial z}(V_2 + V_b) = \begin{bmatrix} 0.02 & 0.03 \\ 0.01 & 0.02 \\ 0.03 & 0.01 \end{bmatrix} = V$

→



14.


14.
Input
Hidden 1
Hidden 2
out
$\hat{y}$
H
V
L_2 loss $= \frac{1}{2}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2$
$\frac{\partial L}{\partial \hat{y}} = -\sum_{i=1}^{m}(y - \hat{y})$
$\frac{\partial \hat{y}}{\partial H} = \hat{y}(1-\hat{y})$
$\frac{\partial H}{\partial V} = h^{(i)}$
$\frac{\partial L}{\partial V} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial H} \cdot \frac{\partial H}{\partial V} = -\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)}) \cdot \hat{y}(1-\hat{y}^{(i)}) \cdot h^{(i)}$