CS M148

Homework 2

KRISH PATEL

Q1) **Perceptron:**

a) Weights are only updated when incorrect predictions are made. Thus only 3 updates.

$$w = 0 + y_1 x_1 + 0 + y_3 x_3 + y_4 x_4$$
$$w = y_1 x_1 + y_3 x_3 + y_4 x_4$$

b) $d = 3$, thus $x_i = [1, x_1, x_2]^T$

$$w = 1[1, 1, 0]^T - 1[1, 1, -3]^T + 1[1, 3, -1]^T$$

$$w = [1, 3, 2]^T$$

Now, the prediction would be $w^T x$

which would be $[1, 3, 2]^T [1, 1, 0]$

$= 1.1 + 3.1 + 2.0 = 1 + 4 + 0$

$= 5$

Sign(5) = 1, thus, it makes a correct prediction on x_1

c) Logistic regression, outputs probabilities, i.e., the value of y strictly lies between 0, and 1, which is interpreted as the probability of the positive class (usually threshold is 0.5). However, for this perceptron, the values are -1 and +1, depending on the sign of the output (A step function) Only makes updates, when the predicted output doesn't match ~~actual output~~ true.

(Q2) a) For the hidden layers, the activations choices were
- ReLU (Different variations include: ReLU, PReLU, ELU, Leaky ReLU)
  - This is simple to implement, especially relu, where gradient is 1 if output of neuron gives a positive output, else 0 if negative.
- Tanh and Sigmoid are also viable choices, they squash the output to [-1, 1] and [0, 1], and are also straightforward to calculate the gradient.

For the binary output layer, the activation function that can be used is a sigmoid.
This is because it outputs probabilies between 0 and 1, and can interpret the output as the likelihood of belonging to one of the classes.

b) $z_1 = w_{11}x_1 + w_{12}x_2 + w_{10}$
$= 0.9 \times 2 + 0.4 \times -3 + 0$
$= 0.6$
Neuron 1 uses RELU, thus $f(z_1) = \underline{0.6}$

$z_2 = w_{21}x_1 + w_{22}x_2 + w_{20}$
$= -1.5 \times 2 - 0.7 \times -3 + 0$
$= -0.9$
$f(z_2) = \frac{1}{1+e^{-(-0.9)}} = \frac{1}{1+e^{0.9}} \approx 0.289$

$\hat{y} = -0.6 \times 0.2 + 1.6 \times 0.289 + 0 \approx 0.3424$
$\uparrow$
w_{30}
≈ 0.34

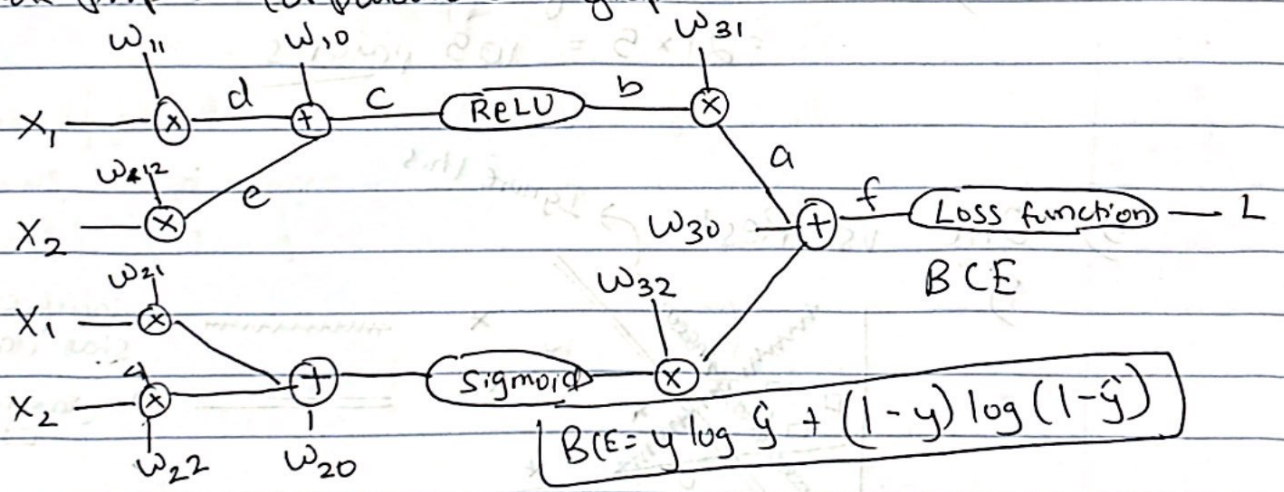
c) Binary cross entropy Loss.

$$L(y,\hat{y}) = \frac{-1}{N}\sum_{i=1}^{N}\left(y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right)$$

$N=1$ loss $= 1\log(0.34) + (0)\log(1-0.34)$

$= -(-1.071775)\ldots \simeq 1.07$

d) Back prop → computational graph



$\frac{\partial L}{\partial f}$ = Using formula of BCE : $\frac{\partial L}{\partial \hat{y}} = \frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$

B through add gate : $\frac{\partial L}{\partial a} = \frac{\partial L}{\partial f}$ $\frac{\partial L}{\partial b} = \frac{\partial a}{\partial b}\cdot\frac{\partial L}{\partial a} =$

~~through~~ $\frac{\partial a}{\partial b} = w_{31}$ derivative through relu = 1 (As positive)

Thus $\frac{\partial L}{\partial c} = 1 \times w_{31} \cdot \frac{\partial L}{\partial a}$. $\frac{\partial L}{\partial e} = \frac{\partial L}{\partial c}$ (Add gate)

$$\frac{\partial L}{\partial w_{12}} = \frac{\partial e}{\partial w_{12}} \cdot \frac{\partial L}{\partial e} = x_2 \cdot w_{31} \cdot \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right)$$

Plugging in values : $1.7523\ldots \simeq 1.75$

e) The number of parameters in (b) is

6 (weight terms) + 3 (bias terms) = 9 total params

$W_1 \rightarrow$ 3 params, $W_2 \rightarrow$ 3 params

$W_3 \rightarrow$ 3 params.

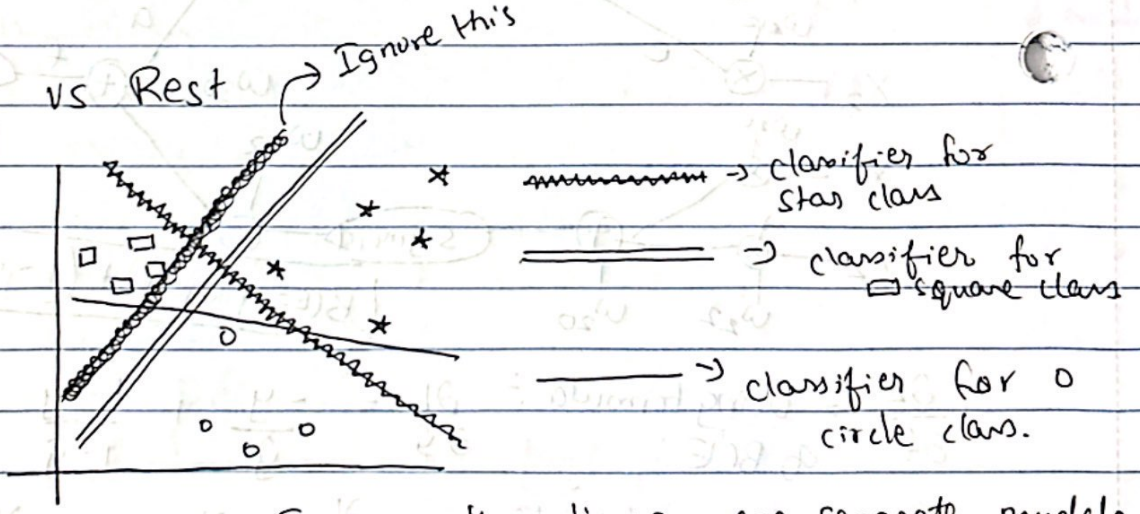
Q3) Multi-class Classification

a) One Vs Rest → Train K binary classifiers, where each model has two classes → probability that it is class k, and probability it is not class K.

Total number of params for 1 classifier:
20 (for 20 features (size of w)) and 1 for bias term.
Thus, total number of params for K=5.
= 21 × 5 = 105 params.

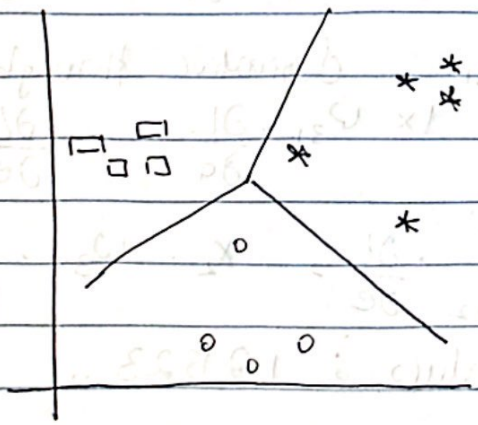
b) One vs Rest

i)



Each of these lines are separate models.

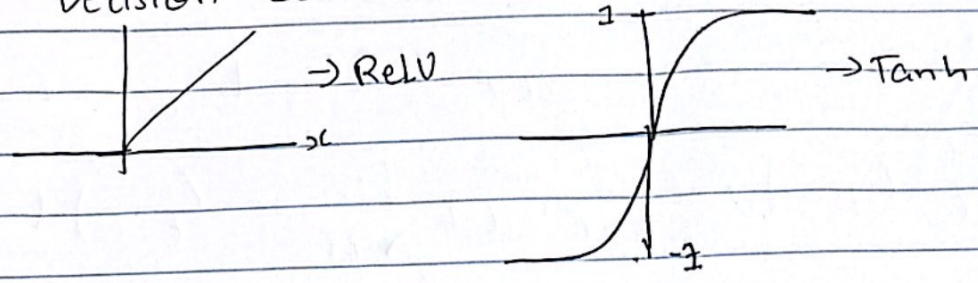
ii)



Multinomial linear boundary

Decision Boundaries

(Q4)



From the above diagrams, we can see that ReLU has a linear activation (~~form~~ when $x>0$) while tanh is more smoother.) (considering that a neural network pieces together different neurons based on activation functions, Tanh would give smoother transitions from one decision boundary of a neuron to another, while relu would look like linear boundaries pieced together. thus, ~~dtg~~ diagram (a) uses a tanh activation function, while (b) uses ReLU activation function.
((a) would look like 10 smooth functions pieced together.)
((b) would look like 10 ~~8~~ linear functions pieced together)