Image model theory - Cat recognition

Max Cotton

1 Setup

- Load image datasets from '.h5' files. (Cat image datasets sourced from https://github.com/marcopeix)
  - Where each image in the input dataset, is represented by an R, G and B matrice, to store the RGB values of each pixel in the 64x64 pixel images
  - Each image's RGB matrices are then 'flattened' into a 1 dimensional array of values, where each element is also divided by 255 (max RGB value) to a number between 0 and 1, to standardize the dataset
  - The output dataset is also loaded, and is reshaped into a 1 dimensional array of 1s and 0s, to store the output of each image (1 for cat, 0 for non cat)
- There is a training dataset with 209 pictures and a test dataset with 50 pictures (fewer pictures are needed for testing)
- Afterwards, the weights and the bias are all initialised to zero/s

2 Forward Propagation

For each epoch, the input array (X) of RGB values is dot producted with the weights (W), summed with a bias (B) (to produce Z) and transferred through the sigmoid function to obtain a prediction between 0 and 1 (A) (where a prediction greater than or equal to 0.5 predicts cat)

3 Back Propagation

- Once a prediction is obtained, you then move back through the network adjusting the weights and the bias
- The "Loss" (L) (how wrong the prediction is) can be calculated with the Loss function, which calculates the average difference between the prediction and the actual value (Y), the average is found by summing the result for each prediction then dividing by the number of predictions (m). This shows how well the network is performing.
  - Where $L = -(\frac{1}{m}) * \sum(Y * log(A) + (1 - Y) * log(1 - A))$ and "log()" is the natural logarithm (e is the base)

- The weights and the bias are adjusted via Gradient Descent, which aims to get the minimum Loss value, with the following formulae:
  - $W = W - learningRate * \frac{\partial L}{\partial W}$
    - Where $\frac{\partial L}{\partial W} = X * (\frac{1}{m})(A - Y)$
  - $B = B - learningRate * \frac{\partial L}{\partial B}$
    - Where $\frac{\partial L}{\partial B} = (\frac{1}{m})(A - Y)$

4 Derivations for $\frac{\partial L}{\partial W}$ and $\frac{\partial L}{\partial B}$

- Functions used so far:
  1. $Z = W.X + B$
  2. $A = \frac{1}{1+e^{-Z}}$
  3. $L = -(\frac{1}{m}) * \sum(Y * log(A) + (1 - Y) * log(1 - A))$
- $\frac{\partial L}{\partial W}$ derivation:
  - By the Chain rule, $\frac{\partial L}{\partial W} = \frac{\partial L}{\partial A} * \frac{\partial A}{\partial Z} * \frac{\partial Z}{\partial W}$
  - Using function 3, $\frac{\partial L}{\partial A} = (-\frac{1}{m})(\frac{Y-A}{A*(1-A)})$
  - Using function 2, $\frac{\partial A}{\partial Z} = A * (1 - A)$
  - Using function 1, $\frac{\partial Z}{\partial W} = X$
  - =¿ $\frac{\partial L}{\partial W} = (-\frac{1}{m})(\frac{Y-A}{A*(1-A)}) * A * (1 - A) * X$
    $= X * (\frac{1}{m})(A - Y)$
- $\frac{\partial L}{\partial B}$ derivation:
  - By the Chain rule, $\frac{\partial L}{\partial B} = \frac{\partial L}{\partial A} * \frac{\partial A}{\partial Z} * \frac{\partial Z}{\partial B}$
  - Derived above, $\frac{\partial L}{\partial A} = (-\frac{1}{m})(\frac{Y-A}{A*(1-A)})$
  - Derived above, $\frac{\partial A}{\partial Z} = A * (1 - A)$
  - Using function 1, $\frac{\partial Z}{\partial B} = 1$
  - =¿ $\frac{\partial L}{\partial W} = (-\frac{1}{m})(\frac{Y-A}{A*(1-A)}) * A * (1 - A)$
    $= (\frac{1}{m})(A - Y)$