CS 188 Summer 2019 — Section Handout 14

Q1. Logistic Regression

We would like to classify some data. We have N samples, where each sample consists of a feature vector $\mathbf{x} = \{x_1, \cdots, x_k\}$ and a label $y = \{0, 1\}$.

We introduce a new type of classifier called logistic regression, which produces predictions as follows:

$$P(Y = 1|X) = h(\mathbf{x}) = s\left(\sum_i w_i x_i\right) = \frac{1}{1 + \exp(-(\sum_i w_i x_i))}$$

$$s(\gamma) = \frac{1}{1 + \exp(-\gamma)}$$

where $s(\gamma)$ is the logistic function, $\exp x = e^x$, and $\mathbf{w} = \{w_1, \cdots, w_k\}$ are the learned weights.

Let's find the weights w_j for logistic regression using stochastic gradient descent. We would like to minimize the following loss function for each sample:

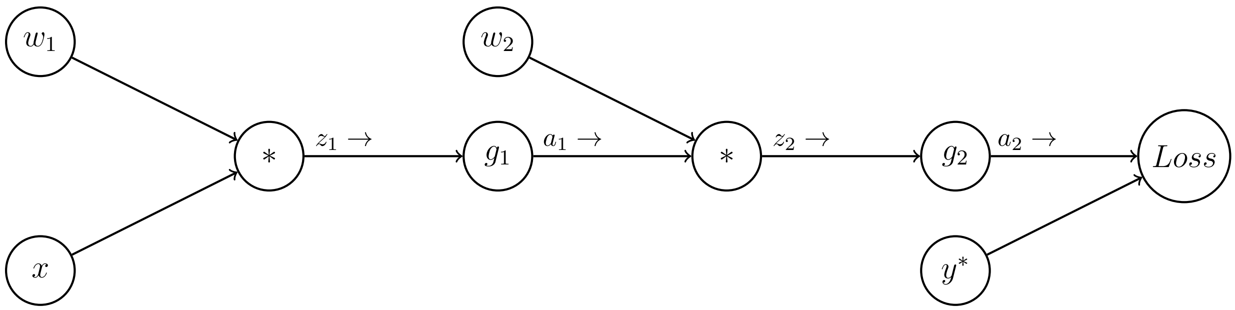
$$L = -[y \ln h(\mathbf{x}) + (1 - y) \ln(1 - h(\mathbf{x}))]$$

(a) Find dL/dw_i. Hint: $s'(\gamma) = s(\gamma)(1 - s(\gamma))$.

(b) Write the stochastic gradient descent update for w_i. Our step size is η.

2 Neural Nets

Consider the following computation graph for a simple neural network for binary classification. Here x is a single real-valued input feature with an associated class y^* (0 or 1). There are two weight parameters w_1 and w_2, and non-linearity functions g_1 and g_2 (to be defined later, below). The network will output a value a_2 between 0 and 1, representing the probability of being in class 1. We will be using a loss function $Loss$ (to be defined later, below), to compare the prediction a_2 with the true class y^*.



1. Perform the forward pass on this network, writing the output values for each node z_1, a_1, z_2 and a_2 in terms of the node's input values:

2. Compute the loss $Loss(a_2, y^*)$ in terms of the input x, weights w_i, and activation functions g_i:

3. Now we will work through parts of the backward pass, incrementally. Use the chain rule to derive $\frac{\partial Loss}{\partial w_2}$. Write your expression as a product of partial derivatives at each node: i.e. the partial derivative of the node's output with respect to its inputs. (Hint: the series of expressions you wrote in part 1 will be helpful; you may use any of those variables.)

4. Suppose the loss function is quadratic, $Loss(a_2, y^*) = \frac{1}{2}(a_2 - y^*)^2$, and g_1 and g_2 are both sigmoid functions $g(z) = \frac{1}{1+e^{-z}}$ (note: it's typically better to use a different type of loss, *cross-entropy*, for classification problems, but we'll use this to make the math easier).

   Using the chain rule from Part 3, and the fact that $\frac{\partial g(z)}{\partial z} = g(z)(1 - g(z))$ for the sigmoid function, write $\frac{\partial Loss}{\partial w_2}$ in terms of the values from the forward pass, y^*, a_1, and a_2:

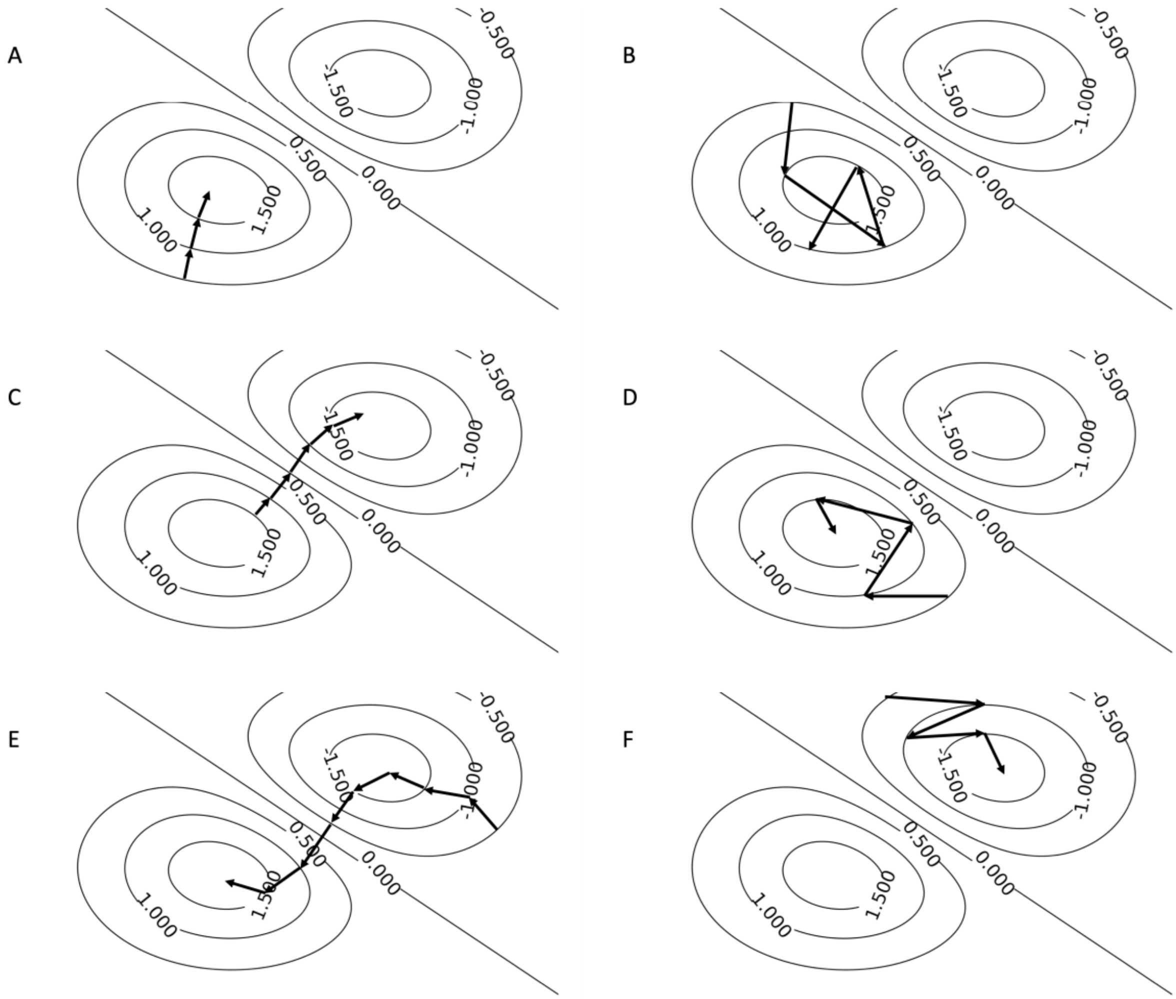
5. Now use the chain rule to derive $\frac{\partial Loss}{\partial w_1}$ as a product of partial derivatives at each node used in the chain rule:

6. Finally, write $\frac{\partial Loss}{\partial w_1}$ in terms of x, y^*, w_i, a_i, z_i:

7. What is the gradient descent update for w_1 with step-size α in terms of the values computed above?

Q3. Gradient Ascent Trajectory

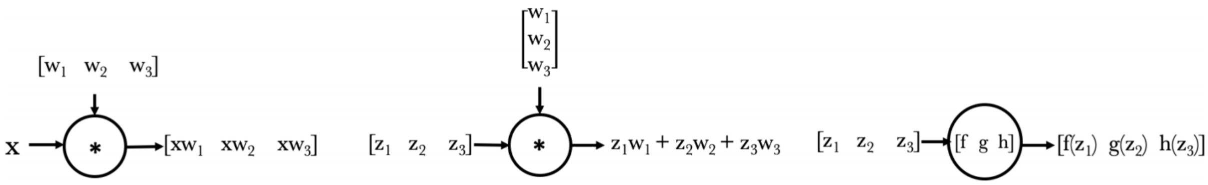
(a) Which of the following paths is a feasible trajectory for the gradient ascent algorithm?



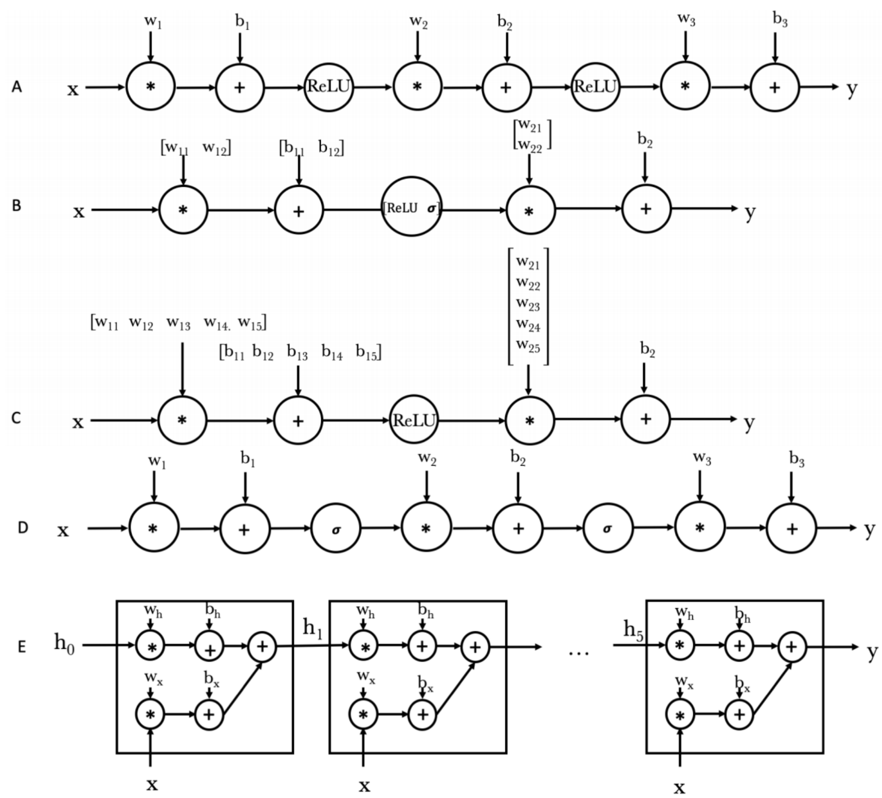
☐ A ☐ B ☐ C ☐ D ☐ E ☐ F

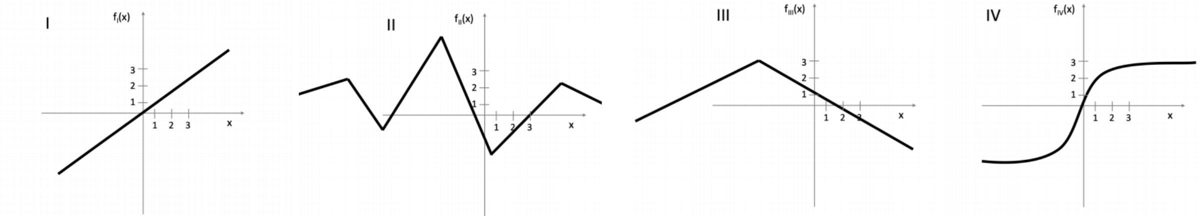
Q4. Neural Networks Representation

(a) We are given the following 5 neural networks (NN) architectures. The operation $*$ represents the matrix multiplication operation, $[w_{i1} \ldots w_{ik}]$ and $[b_{i1} \ldots b_{ik}]$ represents the weights and the biases of the NN, the orientation (vertical and horizontal) is just for consistency in the operations. The term [ReLU σ] in B means applying a ReLU activation to the first element of the vector and a sigmoid (σ) activation to the second element. These operations are depicted in the following figures:



Which of the following neural networks can represent each function?





(i) $\mathrm{f_I}(x)$:

☐ A ☐ B ☐ C ☐ D ☐ E

(ii) $\mathrm{f_{II}}(x)$:

☐ A ☐ B ☐ C ☐ D ☐ E

(iii) $\mathrm{f_{III}}(x)$:

☐ A ☐ B ☐ C ☐ D ☐ E

(iv) $\mathrm{f_{IV}}(x)$:

☐ A ☐ B ☐ C ☐ D ☐ E