Problem 1. (Logistic loss properties)
In class, we developed the logistic loss for binary classification as

$$L(w) = \frac{1}{N}\sum_{i=1}^{N} y^i \log(1+e^{-z^i}) + (1-y^i)\log(1+e^{z^i}), \tag{0.1}$$

where e is the exponential function, log refers to the natural logarithm, $z^i = w_0 + w_1 x_1^i + \cdots + w_d x_d^i$, and our data pairs are $\{(x^i, y^i)\}_{i=1}^N$ with $x^i \in \mathbb{R}^d, y^i \in \{0,1\}$.

1. Compute the derivative $\nabla L(w) \in \mathbb{R}^{d+1}$ of the logistic loss function. Do you see a similarity and a difference between this derivative and the derivative of the least-squares linear regression?

   Solution. Let $x_0^i = 1, \forall i \in \{1, \cdots, N\}$. We have $\forall j \in \{0, 1, \cdots, d\}$,

$$\frac{\partial L(w)}{\partial w_j} = \sum_{i=1}^{N} \frac{\partial L(w)}{\partial z^i}\frac{\partial z^i}{\partial w_j} \tag{0.2}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left((-y^i)\frac{1}{1+e^{-z^i}}e^{-z^i} + (1-y^i)\frac{1}{1+e^{z^i}}e^{z^i}\right)\frac{\partial z^i}{\partial w_j} \tag{0.3}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left((-y^i)\frac{1}{1+e^{-z^i}}e^{-z^i} + (1-y^i)\frac{1}{1+e^{z^i}}e^{z^i}\right)x_j^i \tag{0.4}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{1+e^{-z^i}} - y^i\right)x_j^i, \tag{0.5}$$

   where (0.2) follows by the chain rule. Concatenating $\frac{\partial L(w)}{\partial w_j}, j \in \{0, 1, \cdots, d\}$ gives $\nabla L(w)$.

   Recall that in the least-squares linear regression, the gradient takes the form,

$$\frac{\partial J(w)}{\partial w_j} = \frac{2}{N}\sum_{i=1}^{N}({x^i}^T w - y^i)x_j^i.$$

   Notice that $\hat{y}^i := \frac{1}{1+e^{-z^i}}$ can be interpreted as the probability of predicting class 1. Hence, $(\hat{y}^i - y^i)$ is the error corresponding to this prediction (verify this by considering $y^i = 0$, or $y^i = 1$). Hence, the gradient of the loss vector $L(w)$ with respect to w_j can be written as: $\frac{1}{N}\sum_{i=1}^{N}(\hat{y}^i - y^i)x_j^i$. We can observe that both derivatives take the form of prediction error times learning parameters, which are then averaged over different samples. The difference lies in the concrete forms of predictions. For logistic regression, it is a logistic function, while for least-squares regression, the prediction is a linear function in the input data.

2. Explain an approach to verify the convexity of $L(w)$.

   Solution. There are two common approaches to checking convexity. In class we only talked about the first approach below but here we provide the second one for completeness.

   Second-order method: Compute the Hessian of $L(w)$ and check that it is positive semi-definite for every w. An easier method is provided below.

   Convexity of composite functions: (optional) The composition of a convex function with an affine function is convex. We can verify that $g(z) := \log(1+e^{-z})$ is convex in $z \in \mathbb{R}$ by

taking the second derivative of this function $\frac{d^2g}{dz^2}$. Furthermore, $z = w_0 + w_1x_1 + \cdots + w_dx_d$ is affine in w. Hence, the composition function $\log(1 + e^{-(w_0+w_1x_1+\cdots+w_dx_d)})$ is convex in w.

3. Now, consider a constant classifier by assuming the only parameter of the logistic regression is w_0. In other words, for all data points, we consider only the constant feature 1. Show that the optimal w_0 is a function of the fraction of positive examples, namely, $w_0^\star$ is a function of $\frac{1}{N}|\{i \,|\, y^i = 1\}|$ (where for a set S, $|S|$ denotes its cardinality).

   Solution. By setting the derivative of the loss vector with respect to decision variable w_0 to zero, we can solve for the optimal w_0. Now, from the answer of the first question,

$$\frac{\partial L(w)}{\partial w_0} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{1+e^{-z^i}} - y^i\right)x_0^i. \tag{0.6}$$

   Since $x_0^i = 1$ and $z^i = w_0$, we have

$$\frac{\partial L(w)}{\partial w_0} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{1+e^{-w_0}} - y^i\right). \tag{0.7}$$

   Set the right hand side of Eq. 0.7 to be zero, we get

$$\frac{1}{1+e^{-w_0}} = \frac{\sum_{i=1}^{N} y^i}{N} = \frac{|\{i \,|\, y^i = 1\}|}{N}. \tag{0.8}$$

   Let $p_0^* = \frac{|\{i \,|\, y^i=1\}|}{N}$. We can solve for the optimal weight as $w_0^* = -\log(\frac{1}{p_0^*} - 1)$.

Problem 2. (Empirical distribution and expectation)
Suppose we flip a (possibly biased) coin 5 times and observe the outcomes

$$(\mathrm{H}, \mathrm{T}, \mathrm{H}, \mathrm{H}, \mathrm{T}).$$

Here "H" stands for heads and "T" for tails. Recall that the *empirical distribution* $\hat{p}$ of a random variable based on a set of samples $\{s_i\}_{i=1}^N$ with $s_i \in \{H, T\}$ for all $i = 1, \ldots, N$, is defined as

$$\hat{p}(s) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}_{\{s_i=s\}}, \quad \text{for each outcome } s \in \{H, T\}.$$

1. Write the empirical distribution $\hat{p}$ of this coin based on the samples.

   Solution. We have $N = 5$, and the outcomes contain 3 heads and 2 tails. Hence,

$$\hat{p}(\mathrm{H}) = \frac{3}{5}, \quad \hat{p}(\mathrm{T}) = \frac{2}{5}.$$

2. Let $f(\mathrm{H}) = 1$ and $f(\mathrm{T}) = 0$. Compute the *empirical expectation* $\mathbb{E}_{s\sim\hat{p}}[f(s)]$ under $\hat{p}$.

   Solution.
   $$\mathbb{E}_{s\sim\hat{p}}[f(s)] = f(\mathrm{H})\,\hat{p}(\mathrm{H}) + f(\mathrm{T})\,\hat{p}(\mathrm{T}) = 1\cdot\frac{3}{5} + 0\cdot\frac{2}{5} = \frac{3}{5}.$$

3. Is it possible for the empirical distribution $\hat{p}$ to assign probability 0 to an outcome that has positive true probability?

   Solution. Yes. If a certain outcome never appears in the observed samples, then its empirical probability $\hat{p}(s)$ will be 0, even though the true (unknown) distribution may assign it a positive probability.

Problem 3. (Neural network)
Consider a neural network

$$f : [-2, 2] \to \mathbb{R}, \quad f(x) = W^{[1]T} g\left(W^{[0]T} x + b^{[0]}\right) + b^{[1]},$$

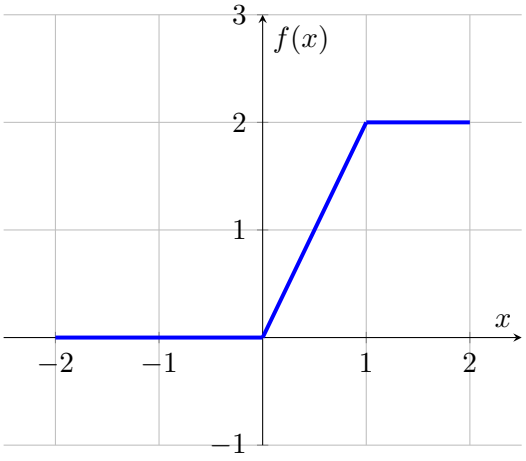
with a single hidden layer and the ReLU activation function[1] $g(x) := \max(0, x)$. Supposing that there are two nodes in the hidden layer, determine the weight matrices $W^{[0]} \in \mathbb{R}^{1\times 2}, W^{[1]} \in \mathbb{R}^{2\times 1}$ and the biases $b^{[0]} \in \mathbb{R}^2, b^{[1]} \in \mathbb{R}$ such that:

1. $f(x) = x$.

   Solution. Since $g(x) - g(-x) = x$, we can choose

   $$W^{[0]} = \begin{bmatrix} 1 & -1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 0 \end{bmatrix}.$$

2. f has the following graph:



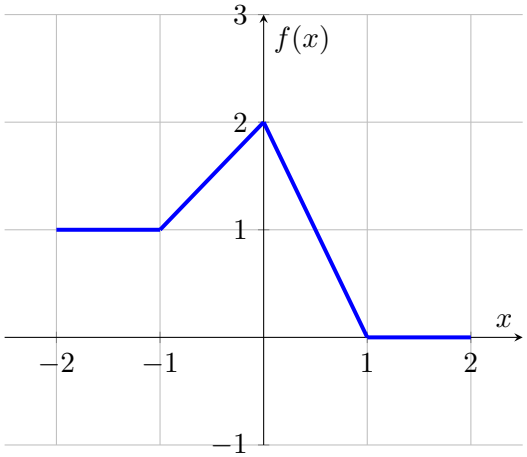
[1] Activation functions are applied elementwise to each node of a hidden layer.

Solution. We can choose

$$W^{[0]} = \begin{bmatrix} 1 & 1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 2 \\ -2 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 0 \end{bmatrix}.$$

Supposing that there are three nodes in the hidden layer, determine the weights $W^{[0]} \in \mathbb{R}^{1\times 3}$, $W^{[1]} \in \mathbb{R}^{3\times 1}$ and the biases $b^{[0]} \in \mathbb{R}^3, b^{[1]} \in \mathbb{R}$ such that:

3. f has the following graph:



Solution. We can choose

$$W^{[0]} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 1 \\ -3 \\ 2 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 1 \end{bmatrix}.$$

Problem 4. (Neural network for regression)
Consider a data set $\{x^i, y^i\}_{i=1}^N$, with $x^i \in \mathbb{R}^d$ and $y^i \in \mathbb{R}^m$. Let our predictor be a neural network $f : \mathbb{R}^d \to \mathbb{R}^m$.

1. Consider a neural network with one hidden layer having 3 nodes and an output layer having m nodes, with activation function $g : \mathbb{R} \to \mathbb{R}$ for each node. Draw this network. How many parameters need to be determined in this network?

   Solution. There are $3(d+1)$ parameters in the first layer, with $3d$ weights and 3 biases. There are $m(3+1)$ parameters in the second layer, with $3m$ weights and m biases.

2. Recall the definition of an affine function from your "Background and notations.pdf" file (posted on Moodle, 8 September - 14 September). Show that if the activation function for each node in the hidden layer and each node in the output layer is the identity, $g(z) = z$, $\forall z \in \mathbb{R}$, then the neural network predictor is the same as a linear predictor with $m(d+1)$ parameters to be determined. Hence, the problem is the same as linear regression.

Solution. Let $W^{[1]} \in \mathbb{R}^{d\times 3}$, $W^{[0]} \in \mathbb{R}^{3\times m}$ denote the weights from the input to the hidden layer and from the hidden layer to the output layer, respectively. Let $b^{[1]} \in \mathbb{R}^3$ and $b^{[0]} \in \mathbb{R}^m$ denote the biases on the hidden layer and the output layer, respectively. Notice that if $g(z) = z$, then the neural network predictor can be written as

$$f(x) = (W^{[0]})^T \left((W^{[1]})^T x + b^{[1]}\right) + b^{[0]} = {W^{[0]}}^T {W^{[1]}}^T x + {W^{[0]}}^T b^{[1]} + b^{[0]},$$

where we are using $x = (x_1, x_2, \ldots, x_d)$. Let $W^T := {W^{[0]}}^T {W^{[1]}}^T \in \mathbb{R}^{m\times d}$, $b := {W^{[0]}}^T b^{[1]} + b^{[0]} \in \mathbb{R}^m$. Then we have $f(x) = W^T x + b$. We can indeed verify this is an affine function since W^T is a matrix and b is a vector.

Note: Recall that for any matrix $M \in \mathbb{R}^{a\times b}$, $f(x) = Mx$ is linear: for any $x, x' \in \mathbb{R}^b$, $\alpha, \beta \in \mathbb{R}$, $f(\alpha x + \beta x') = M(\alpha x + \beta x') = \alpha Mx + \beta Mx' = \alpha f(x) + \beta f(x')$.

3. Now, consider the activation function $g(z) = \tanh(z)$ for each node. Write the predictor f.

Solution. Let $\boldsymbol{W}^{[1]} \in \mathbb{R}^{d\times 3}$ and $\boldsymbol{b}^{[1]} \in \mathbb{R}^3$ be the weights and biases of the hidden layer, and let $\boldsymbol{W}^{[2]} \in \mathbb{R}^{3\times m}$ and $\boldsymbol{b}^{[2]} \in \mathbb{R}^m$ be those of the output layer. The hidden layer output is

$$h = \tanh\left((\boldsymbol{W}^{[1]})^\top x + \boldsymbol{b}^{[1]}\right),$$

where tanh is applied elementwise. The network prediction is then

$$f(x) = (\boldsymbol{W}^{[2]})^\top h + \boldsymbol{b}^{[2]} = (\boldsymbol{W}^{[2]})^\top \tanh\left((\boldsymbol{W}^{[1]})^\top x + \boldsymbol{b}^{[1]}\right) + \boldsymbol{b}^{[2]}.$$

(a) Let $d = 2$, $m = 1$. Suppose all biases are 0 and we pick weights $\hat{\boldsymbol{W}}^{[1]}_{ij} = 1$ for all i, j, and $\hat{\boldsymbol{W}}^{[2]}_{ij} = -1$ for all i, j. For $x^{\text{test}} = [4, -2]^\top$, evaluate $f(x^{\text{test}})$.
Solution. We have one hidden layer with three nodes. First compute the hidden layer activations $h \in \mathbb{R}^3$:

$$h = \tanh((\hat{\boldsymbol{W}}^{[1]})^\top x^{\text{test}}) = \tanh\left(\begin{bmatrix}1 & 1\\ 1 & 1\\ 1 & 1\end{bmatrix}\begin{bmatrix}4\\ -2\end{bmatrix}\right) = \tanh\left(\begin{bmatrix}2\\ 2\\ 2\end{bmatrix}\right) = \begin{bmatrix}\tanh(2)\\ \tanh(2)\\ \tanh(2)\end{bmatrix}.$$

Then the output is

$$f(x^{\text{test}}) = (\hat{\boldsymbol{W}}^{[2]})^\top h = [-1\ -1\ -1]\begin{bmatrix}\tanh(2)\\ \tanh(2)\\ \tanh(2)\end{bmatrix} = -3\tanh(2) \approx -2.892.$$

(b) Write the mean-squared loss function $L(\boldsymbol{W}, \boldsymbol{b})$. What is the domain of this function?
Solution. The mean-squared loss over the dataset $\{(x^i, y^i)\}_{i=1}^N$ is

$$L(\boldsymbol{W}, \boldsymbol{b}) = \frac{1}{N}\sum_{i=1}^{N} \|f_{\boldsymbol{W},\boldsymbol{b}}(x^i) - y^i\|_2^2.$$

The domain of L is the set of all possible weight matrices and bias vectors of the network:

$$\mathrm{dom}(L) = \mathbb{R}^{d\times 3} \times \mathbb{R}^3 \times \mathbb{R}^{3\times m} \times \mathbb{R}^m.$$

(c) Let $\frac{\partial L}{\partial \boldsymbol{W}}$ and $\frac{\partial L}{\partial \boldsymbol{b}}$ denote the partial gradients of the loss with respect to weights and biases, respectively. Write the gradient descent rule for minimizing the loss function L.[2] Would you expect this procedure to find the optimal $\boldsymbol{W}$ and $\boldsymbol{b}$?

Solution. The gradient descent update rules are:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \frac{\partial L}{\partial \boldsymbol{W}}, \qquad \boldsymbol{b} \leftarrow \boldsymbol{b} - \eta \frac{\partial L}{\partial \boldsymbol{b}},$$

where $\eta > 0$ is the learning rate.

Because the network with tanh activations is a non-convex model, the loss function L is generally non-convex. Hence, gradient descent may converge to a local minimum or saddle point, but not necessarily to the global optimum.

[2] You can use $\frac{\partial L}{\partial \boldsymbol{W}}$ and $\frac{\partial L}{\partial \boldsymbol{b}}$ in your expressions without determining them explicitly.