HW2 Liuyi Jin
225009797

1. **UML 2.2**

First, let's focus on the definition of the notation on the two sides of this equation, i.e. $L_{D,f}(h)$ and $L_S(h)$

$$L_{\mathcal{D},f}(h) \stackrel{\text{def}}{=} \mathbb{P}_{x\sim\mathcal{D}}[h(x) \neq f(x)] \stackrel{\text{def}}{=} \mathcal{D}(\{x : h(x) \neq f(x)\}). \quad (1)$$

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m} \quad (2)$$

$$E_{S|_x\sim D^m}[L_S(h)] = E_{S|_x\sim D^m}\left[\frac{|\{i \in [m]: h(x_i) \neq y_i\}|}{m}\right] = \frac{1}{m}E[|\{i \in [m]: h(x_i) \neq y_i\}|] \quad (3)$$

Since$S|_x\sim D^m$, we can interpret it as $x_i\sim D \; and \; y_i = f(x)$ for each$i \in [m]$. With the condition that h is fixed, we can infer that probability $P(h(x_i) \neq y_i)$ is also fixed for each$i \in [m]$. From equation (3), we need to calculate the expectation of the sample size, whose prediction is not equal to the sample output. We denote random variable L as

$$L = \begin{cases} 1, h(x_i) \neq y_i \\ 0, h(x_i) = y_i \end{cases}$$

For each$i \in [m]$. Thus the distribution of L can be regarded as a Bernoulli distribution: $L\sim B\left(m, P\big(h(x) \neq f(x)\big)\right)$. As a result

$$E_{S|_x\sim D^m}[L_S(h)] = E_{S|_x\sim D^m}\left[\frac{|\{i \in [m]: h(x_i) \neq y_i\}|}{m}\right] = \frac{1}{m}E(L) = \frac{1}{m}mP\big(h(x) \neq f(x)\big) = L_{D,f}(h)$$

2. **UML 3.7**

First, we need to make clear of the definition of the Bayes Optimal Predictor.

The Bayes Optimal Predictor.
Given any probability distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$, the best label predicting function from $\mathcal{X}$ to $\{0,1\}$ will be

$$f_{\mathcal{D}}(x) = \begin{cases} 1 & \text{if } \mathbb{P}[y = 1|x] \geq 1/2 \\ 0 & \text{otherwise} \end{cases}$$

Since we are targeting our equation on the distribution on from X to {0, 1}, let's define our loss function as follows,

$$L = \sum_X \sum_Y P(X,Y)I(g(X) \neq Y) = \sum_X P(X) \sum_Y P(Y|X)I(g(X) \neq Y)$$

For any binary classifier, which means from X to {0, 1}, by ignoring the $\sum_X P(X)$ term, Bayes predictor always tends to set the prediction label to be the one with higher probability, which prevents the predictor from generating prediction labels different from the real one with higher probability. In

another word, the Bayes predictor minimizes the probabilities in the process of generating prediction labels.

3. UML 9.5

To stop a perceptron learning process, we need to have

$$y_i \boldsymbol{w}^{*T} \boldsymbol{x}_i > 0, \forall \boldsymbol{x}_i, y_i \; in \; the \; sample$$

Suppose initially, we have $\boldsymbol{w}^{(0)} = (0, \dots, 0)^T$. In each update process, we implement $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + y_i \boldsymbol{x}_i$. When the iteration stops after t[th] iteration, the following equation holds.

$$\boldsymbol{w}^{(t)} = \boldsymbol{w}^{(t-1)} + y_i \boldsymbol{x}_i = \boldsymbol{w}^{(t-2)} + y_i \boldsymbol{x}_i + y_i \boldsymbol{x}_i = \cdots = \sum_{i=1}^{t-1} y_i \boldsymbol{x}_i + \boldsymbol{w}^{(0)} = \sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \quad (1)$$

Before the t[th] update, $\exists i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i < 0$. After t[th] update, $\forall i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i > 0$. This means that

$$\forall i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i = \left(\sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \right) y_i \boldsymbol{x}_i > 0$$

Let's suppose we have a variable $\xi > 0, where \; \eta = \frac{\xi}{n-1}$, then we have the following inequality derived from the inequality above holds.

$$\forall i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i = \xi \left(\sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \right) y_i \boldsymbol{x}_i = \left(\sum_{i=1}^{t-1} \eta y_i \boldsymbol{x}_i \right) y_i \boldsymbol{x}_i > 0 \quad (2)$$

$$\boldsymbol{w}^{(t)} = \xi \left(\sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \right) = \sum_{i=1}^{t-1} \eta y_i \boldsymbol{x}_i = \boldsymbol{w}^{(t-1)} + \eta y_i \boldsymbol{x}_i \quad (3)$$

Given the equation above, we can conclude from inequality (2) that in the new update scheme, after t[th] update, iterations can force the perceptron to converge. This is indicated from the fact that after t[th] iteration, $\forall i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i > 0$. From the comparison of equation (1) and (3), we can see that two perceptron point to the same direction after t[th] iteration.

From equation (1), we have seen that before the t[th] update, $\exists i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i < 0$. This is the same in the new update scheme, $\exists i, s.t. \boldsymbol{w}^{(t)} y_i \boldsymbol{x}_i = \xi \left(\sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \right) y_i \boldsymbol{x}_i = \left(\sum_{i=1}^{t-1} \eta y_i \boldsymbol{x}_i \right) y_i \boldsymbol{x}_i < 0$.

Thus, we have proven this question.

4. BRML 17.3

(1)

$$P(c|\boldsymbol{X}) = \frac{e^{\boldsymbol{w}_c^T \boldsymbol{X}}}{e^{\boldsymbol{w}_1^T \boldsymbol{X}} + e^{\boldsymbol{w}_2^T \boldsymbol{X}}} = \frac{1}{1 + e^{-(\boldsymbol{w}_c^T - \boldsymbol{w}_{3-c}^T)\boldsymbol{X}}}$$

(2)

$$L = \log \prod_{i}^{N} P(c_i|\boldsymbol{X_i}) = \sum_{i=1}^{N} \log P(c_i|\boldsymbol{X_i}) = \sum_{i=1}^{N} \log \frac{e^{\boldsymbol{w}_{c_i}^T \boldsymbol{X_i}}}{\sum_{c'=1}^{C} e^{\boldsymbol{w}_{c_i'}^T \boldsymbol{X_i}}} = \sum_{i=1}^{N} \boldsymbol{w}_{c_i}^T \boldsymbol{X_i} - \sum_{i=1}^{N} \log \left(\sum_{c'=1}^{C} e^{\boldsymbol{w}_{c_i'}^T \boldsymbol{X_i}} \right) \quad (4)$$

(3)

Before we prove that the Hessian matrix of L is negative semidefinite, we define the following function

$$f(\boldsymbol{X}) = \log \left(\sum_{c'=1}^{C} e^{\boldsymbol{w}_{c_i'}^T \boldsymbol{X}} \right) \quad (5)$$

Put the equation (5) into (4), then

$$L = \sum_{i=1}^{N} \boldsymbol{w}_{c_i}^T \boldsymbol{X_i} - \sum_{i=1}^{N} f(\boldsymbol{X_i}) \quad (6)$$

Now, we should note that $f(\boldsymbol{X_i})$ is a log-sum-exp function. As mentioned in Boyd's Convex Optimization book (Chapter 3, page 74), the Hessian of the log-sum-exp function$f(\boldsymbol{X})$ has the following form,

$$\nabla^2 f(x) = \frac{1}{(\mathbf{1}^T v)^2} \left((\mathbf{1}^T v) diag(v) - v v^T \right)$$

Where $v = \left(e^{\boldsymbol{X_1}}, \dots, e^{\boldsymbol{X_n}} \right)$. Then for any given z, we can get

$$z^T \nabla^2 f(x) z = \frac{1}{(\mathbf{1}^T v)^2} \left(\left(\sum_{i=1}^{n} z_i \right) \left(\sum_{i=1}^{n} v_i^2 z_i \right) - \left(\sum_{i=1}^{n} v_i z_i \right)^2 \right)$$

With the Cauchy-Schwarz inequality$(a^T a)(b^T b) \geq (a^T b)^2$, we can confirm that

$$z^T \nabla^2 f(x) z = \frac{1}{(\mathbf{1}^T v)^2} \left(\left(\sum_{i=1}^{n} z_i \right) \left(\sum_{i=1}^{n} v_i^2 z_i \right) - \left(\sum_{i=1}^{n} v_i z_i \right)^2 \right) \geq 0$$

This means that the Hessian of the log-sum-exp function$f(\boldsymbol{X})$ is positive semidefinite for all z, thus $f(\boldsymbol{X_i})$ is a convex function for each *i*. In equation (6), $\sum_{i=1}^{N} f(\boldsymbol{X_i})$ is a positive linear combination of convex functions, with a negative sign before it, and considering$\sum_{i=1}^{N} \boldsymbol{w}_{c_i}^T \boldsymbol{X_i}$ is a positive linear combinations of concave so L is also a concave functions. With this in mind, it is trivial to know that the Hessian matrix of L is negative semidefinite.

5. BRML 17.10

(1)

$$P(c = 1|\boldsymbol{X}) = \sigma \left(b_0 + v_1 g(w_1^T X + b_1) + v_2 g(w_2^T X + b_2) \right)$$

For simplicity, let's denote

$$q^n = b_0 + v_1 g(w_1^T X^n + b_1) + v_2 g(w_2^T X^n + b_2)$$

HW2 Liuyi Jin
225009797

$$P(c=1|\boldsymbol{X^n}) = \frac{e^{q^n}}{1+e^{q^n}}$$

$$L = \log\prod_{n=1}^{N} P(c=1|\boldsymbol{X^n}) = \sum_{n=1}^{N}\log P(c=1|\boldsymbol{X^n}) = \sum_{n=1}^{N}\log\frac{e^{q^n}}{1+e^{q^n}}$$

(2)

$$\frac{\partial L}{\partial \boldsymbol{w_1}} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial \boldsymbol{w_1}} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial \boldsymbol{w_1}} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[-v_1 e^{-\frac{\left(\boldsymbol{w_1^T}X^n+b_1\right)^2}{2}}\left(\boldsymbol{w_1^T}X^n+b_1\right)X^n\right]$$

$$\frac{\partial L}{\partial \boldsymbol{w_2}} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial \boldsymbol{w_2}} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial \boldsymbol{w_2}} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[-v_2 e^{-\frac{\left(\boldsymbol{w_2^T}X^n+b_2\right)^2}{2}}\left(\boldsymbol{w_2^T}X^n+b_2\right)X^n\right]$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial b_1} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial b_1} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[-v_1 e^{-\frac{\left(\boldsymbol{w_1^T}X^n+b_1\right)^2}{2}}\left(\boldsymbol{w_1^T}X^n+b_1\right)\right]$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial b_2} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial b_2} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[-v_2 e^{-\frac{\left(\boldsymbol{w_2^T}X^n+b_2\right)^2}{2}}\left(\boldsymbol{w_2^T}X^n+b_2\right)\right]$$

$$\frac{\partial L}{\partial v_1} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial v_1} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial v_1} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[e^{-\frac{\left(\boldsymbol{w_1^T}X^n+b_1\right)^2}{2}}\right]$$

$$\frac{\partial L}{\partial v_2} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial v_2} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial v_2} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\left[e^{-\frac{\left(\boldsymbol{w_2^T}X^n+b_2\right)^2}{2}}\right]$$

$$\frac{\partial L}{\partial b_0} = \frac{\partial L}{\partial q^n}\frac{\partial q^n}{\partial b_0} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)\frac{\partial q^n}{\partial b_0} = \sum_{n=1}^{N}\left(1+e^{q^n}\right)$$

(3)

When using logistic regression, we are mostly working on getting numerical solutions with local optimization, however, in this model, we have two hidden units, each is implementing logistic regression, we are more likely to get the global optimization if the weights are initialized randomly. Randomly initializing the weights at small values around 0 is also what we usually do when implementing neural network training.

In addition, logistic regression is a generalized real linear model, however, this model with a quadratic function g(x) becomes nonlinear

(4)

Suppose the x variable is one dimensional, then we have two linear lines functions fed into exponential function. We can observe from the figure below, two blue lines are the two linear functions. The outputs values of blue lines are fed into exponential function, which produce the two arch yellow lines. The summation gives the dotted yellow line. Considering the properties of logistic function, this scheme

separate the space into two parts, green space and non-green space. The decision boundary of two dimension is sort of like the figure on the right side.