Introduction to Machine Learning (CSCI-UA.473): Homework 2

Instructor: Sumit Chopra

October 14, 2021

Submission Instructions

You must typeset the answers using IATEXand compile them into a single PDF file. Name the pdf file as: ⟨ Your-NetID ⟩-hw2.pdf. For the programming part of the assignment, complete the Jupyter notebook named HW2.ipynb. Create a ZIP file containing both the PDF file and the completed Jupyter notebook. Name it ⟨ Your-NetID ⟩-hw2.zip. Submit the ZIP file on Brightspace. The due date is October 12^{th} , **2021**, 11:59 PM.

Theory

Question T1: Model Selection (5 points)

Consider that we are learning a logistic regression M^1 and a support vector machine M^2, and we have partitioned the data into three subsets: D_{train} (training set), D_{val} (validation set), and D_{test} test set. The two models are iteratively optimized on D_{train} over T steps, and now we have T logistic regression parameter configurations (i.e., weights and biases) $M_1^1, M_2^1, \ldots, M_T^1$ and T support vector configurations $M_1^2, M_2^2, \ldots, M_T^2$ all with different parameters. We now evaluate the expected cost for all the $2T$ models on training set, validation set, and test set. Thus we have $6T$ quantities $\mathcal{L}^i_{\text{train },t}, \mathcal{L}^i_{val,t}$, and $\mathcal{L}^i_{\text{test},t}$ where $i \in \{1, 2\}$ and $t \in \{1, 2, \ldots, T\}$

1. Which i and t should we pick as the best model and why? (2.5 points)

   Answer: The best model is the one with the lowest weights and biases. For small values of i (at the beginning of the fitting), the model will have higher validation error; while with too many iterations, the model will have higher generalization error due to over-fitting.

2. How should we report the generalization error of the model? (2.5 points)

Answer: For each model M^i where $i \in \{1, 2\}$, we compute the error on the test set as the average of the errors on the test set.

$$\mathbb{E}[\mathcal{L}^i_{\text{test}}] = \frac{1}{T} \sum_{t=1}^{T} \mathcal{L}^i_{\text{test},t}$$

Question T2: Gradient of Multi-Class Logistic Regression (10 points)

The loss function on a single sample (x, y) for a logistic regression model with parameters w for the multi-class classification problem can be written as

$$\mathcal{L}_w(x, y) = -\sum_{j=1}^{K} y_j \cdot \log p_j,$$

where K is the number of classes, y_j is the ground truth label corresponding to the j-th class for the current sample, and p_j is defined as:

$$\begin{aligned} p_j &= \sigma\left(w^T \cdot x\right)_j \\ &= \frac{e^{w_j^T \cdot x}}{\sum_{j=1}^{K} e^{w_j^T \cdot x}} \end{aligned}$$

The function $\sigma()$ is also called the Softmax and the loss function $\mathcal{L}_w$ is called the cross-entropy loss: by far the most popular loss function used to solve multiclass classification tasks.

Compute the gradient of the above loss function with respect to the parameter vector w. Show all the steps of the derivation.

Answer: First compute the gradient of the Softmax function p_j with respect to the parameter w_i and w_j where $i \neq j$:

$$\begin{aligned} \frac{\partial p_j}{\partial w_j} &= \frac{\partial}{\partial w_j} \frac{e^{w_j^T \cdot x}}{\sum_{j=1}^{K} e^{w_j^T \cdot x}} \\ &= \frac{\frac{\partial e^{w_j^T \cdot x}}{\partial w_j} \cdot \sum_{j=1}^{K} e^{w_j^T \cdot x} - e^{w_j^T \cdot x} \cdot \sum_{j=1}^{K} \frac{\partial e^{w_j^T \cdot x}}{\partial w_j}}{[\sum_{j=1}^{K} e^{w_j^T \cdot x}]^2} \\ &= \frac{x e^{w_j^T \cdot x} \cdot \sum_{j=1}^{K} e^{w_j^T \cdot x} - e^{w_j^T \cdot x} \cdot \sum_{j=1}^{K} x e^{w_j^T \cdot x}}{[\sum_{j=1}^{K} e^{w_j^T \cdot x}]^2} \\ &= x \frac{e^{w_j^T \cdot x}}{\sum_{j=1}^{K} e^{w_j^T \cdot x}} - x \frac{e^{w_j^T \cdot x} \cdot \sum_{j=1}^{K} e^{w_j^T \cdot x}}{[\sum_{j=1}^{K} e^{w_j^T \cdot x}]^2} \\ &= x p_j - x p_j^2 \end{aligned}$$

$$
\begin{aligned}
\frac{\partial p_j}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{e^{w_j^T \cdot x}}{\sum_{j=1}^K e^{w_j^T \cdot x}} \\
&= e^{w_j^T \cdot x} \frac{\partial \left(\sum_{j=1}^K e^{w_j^T \cdot x} \right)^{-1}}{\partial w_i} \\
&= -x e^{w_j^T \cdot x} e^{w_i^T \cdot x} \left(\sum_{j=1}^K e^{w_j^T \cdot x} \right) \left(\sum_{i \neq j}^K e^{w_i^T \cdot x} \right) \\
&= -x p_j p_i
\end{aligned}
$$

The gradient of the loss function with respect to the parameter vector w is given by:

$$
\begin{aligned}
\boldsymbol{\nabla}_w \mathcal{L}_w(x, y) &= -\frac{\partial}{\partial w_j} \sum_{i=1}^K y_i \cdot \log p_i \\
&= -\sum_{i=1, i \neq j}^K y_i \frac{\partial \log p_i}{\partial w_j} - y_j \frac{\partial \log p_j}{\partial w_j} \\
&= -\sum_{i=1, i \neq j}^K y_i p_i^{-1} \frac{\partial p_i}{\partial w_j} - y_j p_j^{-1} \frac{\partial p_j}{\partial w_j} \\
&= \sum_{i=1, i \neq j}^K y_i p_i^{-1} x p_j p_i - y_j p_j^{-1} (x p_j - x p_j^2) \\
&= \sum_{i=1, i \neq j}^K x y_i p_j + x y_j p_j - x y_j \\
&= x p_j \sum_{i=1}^K y_i - x y_j \\
&= x p_j - x y_j
\end{aligned}
$$

Question T3: Maximum Likelihood Estimate of a Gaussian Model (10 Points)

Assume you are given a dataset $\mathcal{D}$ of n real numbers $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathrm{Re}, \forall i$. Derive the maximum likelihood estimate of the mean μ and variance σ, of the 1 -dimensional Gaussian distribution. Note that μ and σ are the learnable parameters.

1. Write down the expression of the log-likelihood $\mathcal{L}_{\mu,\sigma}(\mathcal{D})$ of the data set $\mathcal{D}$ as a function of μ and σ. (2 points)

2. Compute the partial derivative of $\mathcal{L}_{\mu,\sigma}(\mathcal{D})$ with respect to μ, equate to zero and solve for μ. (4 points)

3. Compute the partial derivative of $\mathcal{L}_{\mu,\sigma}(\mathcal{D})$ with respect to σ, equate to zero and solve for σ. (4 points)

Question T4: Hinge loss gradients (5 points)

Unlike the Cross-Entropy loss, the Hinge loss (defined below), is not differentiable everywhere with respect to the parameters θ :

$$\mathcal{L}_{\text{Hinge}}\left(x, y, \theta\right) = \max\left[0, 1 - y \cdot f_\theta(x)\right],$$

for some parametric function f_θ. Does it mean that we cannot use a gradient-based optimization algorithm for finding a solution that minimizes the hinge loss? If not, what can we do about it?

Practicum

See the accompanying Python notebook.

Question P1: Metrics for a binary classifier (20 points)

Question P2: Support Vector Machines (50 points)