Homework 1 40pt (Math 466, Spring 2025)

Due Date: Feb. 06, Thur, 10 pm

Instructions

- Submissions must be made in PDF format. If there are coding problems, please provide source code that directly generates the reported results.
- In written problems, if no derivation is asked for, just give the answer. Otherwise, try to give the derivation in a clear and mathematically rigorous way.
- For coding problems, please include a mini-report summarizing your experiment, along with any relevant figures or tables.

  When creating plots or tables, ensure they are clear and easy to interpret by using different colors or markers, and by adding axis labels, titles, captions, etc. to enhance readability.

  You are also required to submit the code used to generate the results presented in your mini-report. This can be provided as either a .ipynb or .py file.
- Collaboration and AI tools are allowed, but you must write your solutions independently and acknowledge collaborators/AI tools.

Questions

1. **(6pt) This problem is about multiclass classification.** In class, we discussed logistic regression for binary classification. This question extends that discussion to multiclass classification using a similar approach.

   Recap:

   For binary classification, assume $y \in \{-1, 1\}$ and define

   $$p(x) := \mathbb{P}(y = 1|x)$$

   the probability that x belongs to the positive class. Since $p(x) \in [0, 1]$, directly parameterizing this function is challenging. Instead, we parameterize the logit

   $$\log \frac{p(x)}{1 - p(x)}$$

using a linear model $f(x; w, b) = w^\top x + b$, where (w, b) are the parameters. Then we apply **Maximum Likelihood Estimation (MLE)** to determine the optimal parameters (w, b).

Multiclass classification:

Now consider a K-class classification, i.e. $y \in \{1, 2, \cdots, K\}$. Define

$$p_k(x) := \mathbb{P}(y = k|x), \quad k = 1, 2, \cdots, K$$

as the probability that x belongs to the k-th class. To parameterize these probabilities, we parameterize $K - 1$ logit functions:

$$\log \frac{p_k(x)}{p_K(x)}, \quad k = 1, 2, \cdots, K - 1.$$

Assume the k-th logit function is parameterized by $f_k(x; \theta_k)$, where θ_k represents the parameters. Let $\theta = (\theta_1, \theta_2, \cdots, \theta_{K-1})$ denote the collection of all parameters.

(a) (2pt) For $k = 1, 2, \cdots, K$, express $p_k(x; \theta)$ in terms of $\{f_j(x; \theta), j = 1, 2, \cdots, K - 1\}$. (You can drop θ in this question, but keep in mind that the probability $p(x)$ is parameterized by θ.)

(b) (2pt) Assume we are given data $\{(x_i, y_i)\}_{i=1}^N$ with $y_i \in \{1, 2, \cdots, K\}$. Write down the probability of the given data under the parameterized probability $p_k, k = 1, 2, \cdots, K$ in (a).

(c) (2pt) Following the principle of **MLE**, formulate the optimization problem that the optimal parameters θ satisfy. Argue that the problem is equivalent to

$$\min_\theta - \sum_{i=1}^N \log p_{y_i}(x_i; \theta).$$

2. **(14 pt) This problem is about nonlinear regression.** In class, we explored ridge regression under the assumption that the underlying model is linear. However, the linear model assumption can often be too restrictive. In this problem, we investigate how to extend ridge regression to nonlinear models.

Recap:

Consider data $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and assume x_i is centralized. Ridge regression assumes a linear model: $y = w_d^\top x + \epsilon$ where $w_d \in \mathbb{R}^d$ and ϵ represents white noise. The parameter w_d is determined by solving:

$$\min_{w_d \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^N (w_d^\top x_i - y_i)^2 + \frac{\lambda}{2} \|w_d\|_2^2.$$

with $\lambda > 0$ as the regularization coefficient.

Nonlinear models:

For nonlinear models, suppose we can identify a function $\phi : \mathbb{R}^d \to \mathbb{R}^p$ that resolves the nonlinearity such that the transformed data $(\phi(x), y)$ satisfies a linear model: $y = w_p^\top \phi(x_i) + \epsilon, w_p \in \mathbb{R}^p$. We can then apply ridge regression to $(\phi(x_i), y_i)$, solving for w_p as follows:

$$\min_{w_p \in \mathbb{R}^p} \frac{1}{2} \sum_{i=1}^{N} (w_p^\top \phi(x_i) - y_i)^2 + \frac{\lambda}{2} \|w_p\|_2^2. \tag{1}$$

and then predict y by $y = w_p^\top \phi(x)$ for given x.

(Remark: The dimension p can be very large or even infinite. In this problem, we assume p is finite.)

(Remark: Finding the function ϕ explicitly is often impractical. Instead, we leverage the **kernel** trick to circumvent this issue, as discussed in part (c) of this problem.)

(Remark: This approach is called kernel ridge regression and is part of the broader class of kernel methods in machine learning. We will explore kernel methods in greater detail later in the lectures.)

(a) (4pt) Let

$$\mathbf{\Phi} := \begin{bmatrix} \phi(x_1)^\top \\ \phi(x_2)^\top \\ \vdots \\ \phi(x_N)^\top \end{bmatrix} \in \mathbb{R}^{N \times p}, \quad \mathbf{y} := \begin{bmatrix} y_1^\top \\ y_2^\top \\ \vdots \\ y_N^\top \end{bmatrix} \in \mathbb{R}^N.$$

prove that $\lambda \mathbf{I}_p + \mathbf{\Phi}^\top \mathbf{\Phi}$ is invertible and that the optimizer of (1) is given by

$$w_p = (\lambda \mathbf{I}_p + \mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y}, \tag{2}$$

where $\mathbf{I}_p$ is a $p \times p$ identity matrix.

(b) (4pt) Prove the following matrix identity: for any invertible matrices $P \in \mathbb{R}^{n \times n}$, $R \in \mathbb{R}^{m \times m}$, and any matrix $B \in \mathbb{R}^{m \times n}$,

$$(P^{-1} + B^\top R^{-1} B)^{-1} B^\top R^{-1} = P B^\top (B P B^\top + R)^{-1}. \tag{3}$$

Using (2) and (3), show that the optimizer of (1) can also be expressed as:

$$w_p = \mathbf{\Phi}^\top (\mathbf{\Phi}\mathbf{\Phi}^\top + \lambda \mathbf{I}_N)^{-1} \mathbf{y}. \tag{4}$$

(Remark. Equation (3) will be discussed from the perspective of optimization duality later in the course. For now, use linear algebra to prove it.)

(Remark. Solving for w_p using (2) requires inverting a $p \times p$ matrix, while (4) requires inverting an $N \times N$ matrix. When p is large or infinite, (4) is computationally preferable, as N is always finite.)

(c) (2pt) Recall that our goal is to predict y given x. The function ϕ and the parameters w_p are intermediate tools. Define the kernel function: $K(x, \tilde{x}) := \phi(x)^\top \phi(\tilde{x})$. Show that, to predict y using $y = w_p^\top \phi(x)$ and (4), we only need the inner products $\phi(x)^\top \phi(\tilde{x})$, i.e., the kernel $K(x, \tilde{x})$, rather than the explicit formulation of $\phi(x)$.

(d) (4pt) Using the definition of K in part (c), prove that:

- The kernel is symmetric: $K(x, \tilde{x}) = K(\tilde{x}, x)$ for any x and $\tilde{x}$.
- The kernel is positive semidefinite: for any set of $\{x_i \in \mathbb{R}^d\}_{i=1}^N$ and $\{c_i \in \mathbb{R}\}_{i=1}^N$,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j K(x_i, x_j) \geq 0.$$

3. **(20 pt) This problem explores parametric and non-parametric models.**

A parametric model is characterized by a **finite** number of parameters, whereas a non-parametric model assumes that the data distribution cannot be described by a finite set of parameters.

In class, we discussed logistic regression with a linear predictor function (parametric model), k-nearest neighbors (kNN, non-parametric model), and multilayer perceptrons (MLP). In this question, we explore the difference between parametric and non-parametric models and the special pattern of MLP. You will implement these three methods using the `sklearn` package and compare their performance on two different datasets, each with varying numbers of training samples.

For the implementations:

- For logistic regression, use the class `LogisticRegression`.
- For kNN, use the class `KNeighborsClassifier`.
- For MLP, use the class `MLPClassifier`.

For all three models, you may use the default solver parameters provided by `sklearn`.

(a) (2pt) Based on the definition, is MLP a parametric model or a non-parametric model?

(b) (4pt) Use the provided code to generate two classes of linearly separable data. Train the three models on the following training datasets:

- `(X_train1, y_train1)` (100 training samples),
- `(X_train5, y_train5)` (500 training samples),
- `(X_train10, y_train10)` (1000 training samples).

Then, test the models on the same testing dataset `(X_test, y_test)` (100 testing samples). Record the training accuracy and testing accuracy for each method across the different training sample sizes.

(c) (4pt) Use the provided code to generate two-ring data. Repeat part (b) on the two-ring dataset. Specifically:

- Train the three models on `(X_train1, y_train1)` (100 training samples), `(X_train5, y_train5)` (500 training samples), and `(X_train10, y_train10)` (1000 training samples).
- Test the models on the same testing dataset `(X_test, y_test)` (100 testing samples).

Record the training accuracy and testing accuracy for each method across the different training sample sizes.

(d) (6pt) Summarize your observations from parts (b) and (c). To make the comparisons clearer:

- Summarize the accuracies in a table.
- Plot the decision boundaries for each model to visualize differences.

If the results from parts (b) and (c) are insufficient to identify patterns, consider conducting additional experiments with more varied training sizes.

(e) (4pt) Does the performance behavior of MLP resemble that of a parametric model or a non-parametric model? Combining your answer to part (a), how do you understand the performance behavior of MLP?