Labs
Machine Learning Course
Spring 2025

Westlake University
Department of Artificial Intelligence, SOE
Tao Lin & Kaicheng YU
https://github.com/LINs-lab/course_machine_learning

Problem Set Lab 04, March 13, 2025 (Generalization, and Model Selection)

Goals. Goals. The goal of this week's lab is to:

- Focus on the theoretical content from Week 2 until now.
- Implement 4-fold cross-validation.
- Understand the bias-variance decomposition.

Submission instructions:

- Please submit a PDF file (for solutions to the graded labs) to canvas.
- Deadline: 23:59 on Mar. 23, 2025

1 Graded labs [80 pts]

Problem 1 (Subgradients [5 pts]):

Compute a subgradient for the mean average error (MAE) cost function. That is for any $\mathbf{w}$, give a subgradient $\mathbf{g}$ at $\mathbf{w}$. [Hint: you are allowed to assume the gradient of f is known at any $\mathbf{w}$.]

$$\mathcal{L}(\mathbf{w}) = \mathsf{MAE}(\mathbf{w}) := \frac{1}{N}\sum_{n=1}^{N} \left|y_n - f(\mathbf{w}, \mathbf{x}_n)\right|$$

Problem 2 (Maximum-Likelihood [15 pts]):

In class, we saw that the Maximum-Likelihood Estimation of gaussian noise leads to the Mean Square Error cost function. Derive the MLE of a linear model assuming Laplacian noise with location 0 and scale 1.

- (5pts) Our model is $y_n = \mathbf{w}^\top \mathbf{x}_n + \epsilon$, where ϵ follows a Laplace$(0, 1)$ distribution, $\Pr_{(}[\epsilon]) = \frac{1}{2}e^{-|\epsilon|}$.
- (5pts) Find the likelihood $\Pr_{(}[y]_n | \mathbf{x}_n, \mathbf{w})$.
  Hint: if x follows a Laplace distribution of location μ, $\Pr_{(}[x]) = \frac{1}{2}e^{-|x-\mu|}$.
- (5pts) Find the log-likelihood, and compare with the Mean Absolute Error.

Problem 3 (Weighted least-squares [20 pts]):

Suppose we have a regression dataset with N pairs $\{y_n, \mathbf{x}_n\}$ where y_n is a real-valued scalar, $\mathbf{x}_n$ is a real-valued vector of length D, and we wish to fit a linear model $f(\mathbf{x}_n) = \boldsymbol{\beta}^T \tilde{\mathbf{x}}_n$ where $\boldsymbol{\beta}$ is a vector with entries $\beta_0, \beta_1, \ldots, \beta_D$ and $\tilde{\mathbf{x}}_n^T = [1\ \mathbf{x}_n^T]$.

Suppose we minimize the following cost function:

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2}\sum_{n=1}^{N} w_n \left(y_n - \boldsymbol{\beta}^T \tilde{\mathbf{x}}_n\right)^2 \tag{1}$$

where $w_n > 0$ are known real-valued scalars.

1. (5pts) Derive the normal equations for this cost function. You should write an expression in matrix-vector form similar to the expression for least-squares given in the lecture notes.

2. (5pts) Discuss the conditions under which the solution $\boldsymbol{\beta}^*$ is unique.

3. (5pts) Assuming that these conditions hold, write down the expression for the unique solution.

4. (5pts) We showed in the lectures that the least-squares cost function can be derived using a probabilistic model. Derive a probabilistic model under which minimizing the negative of the log-likelihood gives the same solution as the cost function shown above in (1).

Problem 4 (Multiple-Output Regression [20 pts]):

Let $S = \{(\mathbf{y}_n, \mathbf{x}_n)\}_{n=1}^N$ be our training set for a regression problem with $\mathbf{x}_n \in^D$ as usual. But now $\mathbf{y}_n \in^K$, i.e., we have K outputs for each input. We want to fit a linear model for each of the K outputs, i.e., we now have K regressors $f_k(\cdot)$ of the form

$$f_k(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}_k,$$

where each $\mathbf{w}_k^\top = (w_{k1}, \cdots, w_{kD})$ is the weight vector corresponding to the k-th regressor. Let $\mathbf{W}$ be the $D \times K$ matrix whose columns are the vectors $\mathbf{w}_k$.

Our goal is to minimize the following cost function $\mathcal{L}$:

$$\mathcal{L}(\mathbf{W}) = \sum_{k=1}^{K} \sum_{n=1}^{N} \frac{1}{2\sigma_k^2} (y_{nk} - \mathbf{x}_n^\top \mathbf{w}_k)^2 + \frac{1}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|_2^2,$$

where the σ_k are known real-valued scalars. Let $\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_K)$.

For the solution, let $\mathbf{X}$ be the $N \times D$ matrix whose rows are the feature vectors $\mathbf{x}_n$.

1. (5pts) Write down the normal equations for $\mathbf{W}^\star$, the minimizer of the cost function. I.e., what is the first-order condition that $\mathbf{W}^\star$ has to fulfill in order to minimize $\mathcal{L}(\mathbf{W})$.

2. (5 pts) Is the minimum $\mathbf{W}^\star$ unique? Assuming it is, write down an expression for this unique solution.

3. (10 pts) Write down a probabilistic model, so that the MAP solution for this model coincides with minimizing the above cost function. Note that this will involve specifying the the likelihoods as well as a suitable prior (which will give you the regression term).

Problem 5 (Prove the Generalization Gap Bound [20 pts]):

1. (5pts) State Hoeffding's Inequality formally. Explain the meaning of each term in the inequality and its significance in machine learning.

2. (10pts) Derive the expression for ϵ in Hoeffding's Inequality:

$$\mathbb{P}\left(|L(h) - \hat{L}(h)| \geq \epsilon\right) \leq 2e^{-2n\epsilon^2},$$

given a confidence level $1 - \delta$. Show the steps to solve for ϵ in terms of n and δ.

3. (5pts) Explain how the bound ϵ changes as:

   - The number of training samples n increases.
   - The confidence level $1 - \delta$ increases.

2 Non-graded labs

2.1 Cross-validation

Exercise 1:

Implement 4-fold cross-validation.

- In this exercise, please fill in the notebook functions `cross_validation()` and `cross_validation_demo()`, and perform 4-fold cross-validation for polynomial degree 7. Plot the train and test RMSE as a function of λ. The resulting figure should look like Figure 1.
- How will you use 4-fold cross-validation to select the best model among various degrees, say from 2 to 10? Write code to do it in `best_degree_selection()`.

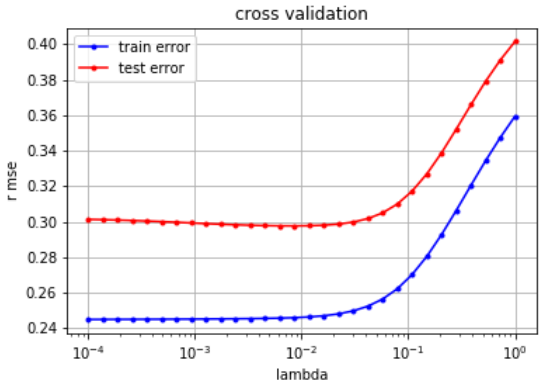

Figure 1: Effect of λ on training and test errors, calculated using 4-fold cross-validation

2.2 Visualizing the Bias-Variance Decomposition

Last lecture we introduced model selection, and we saw that the model complexity is crucial to the performance. In this problem, we will further investigate the effect of *model complexity* with the concept of *bias-variance decomposition*.

We will implement the figures seen in class representing the tradeoff and also seen in Figure 2 : for a big polynomial degree, the bias is small but the variance is large. The opposite is true for a small polynomial degree (however notice that the variance is still quite important). Choosing an intermediate degree leads to consistent predictions which are close to the true function we want to learn (optimal bias / variance tradeoff).

Exercise 2:

Visualizing the bias-variance trade-off.

- Complete the notebook function `bias_variance_one_seed()`: for 15 random datapoints, it finds the optimal fit (using the least square formula, with no regularisation λ) for a polynomial expansion of degree 1, 3 and 6.
- you can play around by changing the seed, the number of datapoints, the degree of the polynomial expansion etc.
- Now complete the notebook function `bias_variance_demo()` which performs many times the previous experiment but with a new random training set each time. You should obtain something similar to Figure 2.
- Comment the figures by explaining how the bias / variance tradeoff is shown in these plots.
- You can play around by changing the function you want to learn, the variance of the gaussian noise σ^2, the degree of the polynomial expansion etc.

- **BONUS:** you can do similar figures but now you fix the degree of the polynomial expansion and add some regularisation λ. You will observe a similar bias / variance tradeoff when changing the magnitude of the regularisation.

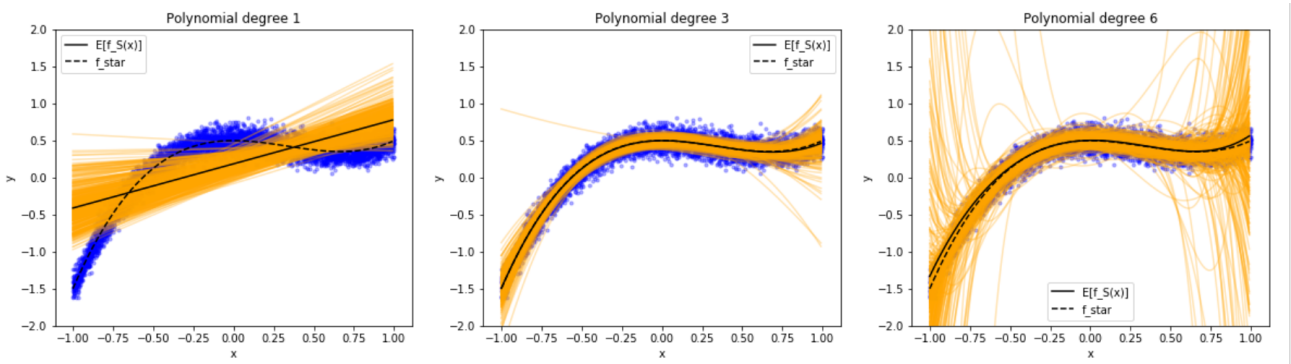

Figure 2: Visualizing the Bias-Variance Trade-off.