Problem 1 [12 marks]

Let $\mathbf{x} \in \mathbb{R}^{N_x}$ be the input features, and $t \in \mathbb{R}$ be the output of a regression task.

The computation of a two-layer perceptron---one of the simplest neural networks---is defined as

$$\mathbf{h} = f(\mathbf{W}_{\text{hid}}\mathbf{x} + \mathbf{b}_{\text{hid}})$$
$$t = \mathbf{w}_{\text{out}}^{\top}\mathbf{h} + b_{\text{out}}$$

where $\mathbf{h} \in \mathbb{R}^{N_h}$ is an intermediate step for computing t and N_h is the dimension of the $\mathbf{h}$ vector. $\mathbf{W}_{\text{hid}} \in \mathbb{R}^{N_h \times N_x}$, $\mathbf{b}_{\text{hid}} \in \mathbb{R}^{N_h}$, $\mathbf{w}_{\text{out}} \in \mathbb{R}^{N_h}$, and $b_{\text{out}} \in \mathbb{R}$ are the parameters. f is a nonlinear function, for example, tanh, applied to every element of its vector input.

Deep neural networks are likely to be overfitting, and suppose we indeed observe that our current two-layer perceptron is overfitting. Name four (4) distinct approaches that can alleviate/prevent overfitting. Each approach should be a new paragraph of one sentence. No detailed explanation is needed.

Solution:

1) Reduce N_h
2) Make f linear, or equivalently, use a linear regression model
3) Add L2 or L1 penalty for W and/or b
4) Use fewer features, i.e., reducing N_x
5) Increase the training dataset
6) K-fold validation if currently using hold-out validation. I may have inconsistent marking criteria here. If you said “cross-validation,” you can interpret it as K-fold validation. You may let me know, and I’ll add marks. If you say validation, then it’s wrong.

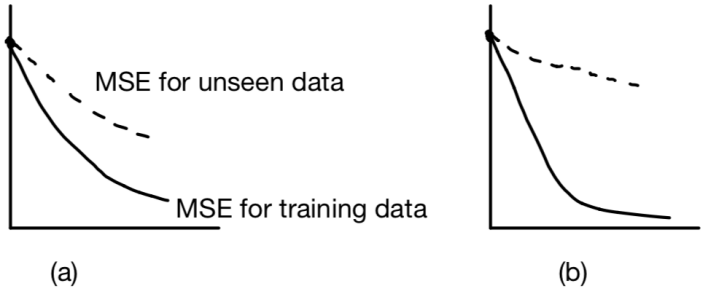
Other solutions not covered in the course are also acceptable as long as they’re correct.

7) Apply dropout
8) Model ensemble
9) Early stopping
10) etc.

Problem 2 [12 marks]

Below are the learning curves of a gradient-based optimization for some machine learning models.

- Solid lines are the curves for training data, and dashed lines are the curves for unseen data
- X-axis (horizontal) is the number of epochs, and y-axis (vertical) is the mean square error.



a) With some tuning of regularization, the training curve goes down, whereas the dashed curve goes up. In other words, the curves change from (a) to (b). Are we putting more regularization or less regularization? (Word limit: 5 words)

b) What if the other way around? In other words, how would the solid and dashed curves change if we put [more | less] regularization? If your answer to a) is more regularization, then here you assume less regularization, and vice versa. (Word limit: 10 words)

Requirement: **No** explanation is needed. Please provide the answer with the below template.

a) ____ regularization

b) Solid: ____; Dashed ____.

Word limit does not count the template words or punctuations. **Exceeding word limit results in 0 mark**.

Solution:

a) Less

b) Up, unknown

Problem 3 [15 marks]

Below is a proof that MLE is equivalent to MSE under certain conditions. Please explain every line of the derivation.

$$\widehat{\mathbf{w}}_{\text{MLE}} = \underset{\mathbf{w}}{\text{argmax}}\, P(\mathcal{D}; w) \tag{1}$$

$$= \underset{\mathbf{w}}{\text{argmax}}\, P(t^{(1)}, t^{(2)}, \cdots, t^{(M)} | \mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(M)}; \mathbf{w}) \tag{2}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \prod_{m=1}^{M} P(t^{(m)} | \mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(M)}; \mathbf{w}) \tag{3}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \prod_{m=1}^{M} P(t^{(m)} | \mathbf{x}^{(m)}; \mathbf{w}) \tag{4}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \log \prod_{m=1}^{M} P(t^{(m)} | \mathbf{x}^{(m)}; \mathbf{w}) \tag{5}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \sum_{m=1}^{M} \log P(t^{(m)} | \mathbf{x}^{(m)}; \mathbf{w}) \tag{6}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \sum_{m=1}^{M} \log \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{1}{2\sigma^2} (t^{(m)} - \mathbf{w}^\top \mathbf{x}^{(m)})^2 \right\} \tag{7}$$

$$= \underset{\mathbf{w}}{\text{argmax}} \sum_{m=1}^{M} \left(-\frac{1}{2\sigma^2} (t^{(m)} - \mathbf{w}^\top \mathbf{x}^{(m)})^2 \right) \tag{8}$$

$$= \underset{\mathbf{w}}{\text{argmin}} \frac{1}{2M} \sum_{m=1}^{M} (t^{(m)} - \mathbf{w}^\top \mathbf{x}^{(m)})^2 \tag{9}$$

Hints:

- Your solution should follow the following format:
  (1) Explanation for line (1)
  (2) Explanation for line (2)
  etc.
- If there's any model assumption, please state the assumption clearly.
- If there's any independence assumption, please explain what variables are independent of what.
- If you use a basic math identity, also state it.
- Crappy notations are accepted, such as w' or w.T for transpose, x^(m) for $\mathbf{x}^{(m)}$. Comment on your notations after providing the solutions.

Solution:

1) Definition of maximum likelihood estimation
2) We model the probability distribution of the target assuming the input is known
3) We assume t^(1), …, t^(M) are independent to each other
4) t^(m) only depends on x^(m), independent of other x^(n), for n not m
5) Log is a monotonic function, which does not change the optimality
6) Log product is the same as sum of log
7) We assume t^(m) is from a Gaussian distribution, i.e., t^(m)~ N(w^T x^(m), sigma^2)
8) log A/B = log A - log B, and the term log[sqrt(2pi) sigma] is a constant. Then log cancels exp
9) Changing the positive multiplicative factor does not affect the optimality. Changing the sign makes argmax to argmin

Problem 4 [31 marks]

[5 marks] Present the ℓ_2-penalized mean square error (MSE) for linear regression. (Dropping constants will not result in mark deduction.)

[10 marks] Derive the closed-form solution.

[5 marks] Suppose we would like to reduce variance and increase bias, what can we do with the ℓ_2-penalized MSE loss?

[11 marks] We know in the lectures that MSE is an unbiased estimate. The Gauss--Markov Theorem further states that, under the same assumptions, MSE yields the best linear unbiased estimate (BLUE), where the best means the least variance.

However, we also learned that ℓ_2-penalty is a tradeoff between bias and variance. But MSE gives no bias and the least variance. Do they contradict each other (1 mark)? And why (10 marks)?

Hints:

Useful matrix calculus identities

- If $\mathbf{A}$ is not a function of $\mathbf{x}$, then $\nabla_{\mathbf{x}}\mathbf{A}\mathbf{x} = \mathbf{A}^{\top}$
- If $\mathbf{A}$ is not a function of $\mathbf{x}$ and $\mathbf{A}$ is symmetric, then $\nabla_{\mathbf{x}}\mathbf{x}^{\top}\mathbf{A}\mathbf{x} = 2\mathbf{A}\mathbf{x}$

Solution:

a)

L2-penalized MSE is
J = || Xw-t ||^2 / 2M + ||w||^2 / 2
where || || is the L2-norm. k is a coefficient balancing the MSE and L2-penalty

Note: Dropping 1/2M is fine, but dropping k or k/2 leads to mark deduction. You have to balance the two terms in a certain way.

b)

Close-form solution:
Compute the partial derivative of J by using the matrix calculus

J = (Xw - t)' (Xw - t) / 2M + k w'w / 2
= (w' X' X w - 2 t' X w + t't)/2M + k w'w /2

dJ / dw = (2 X'X w - 2 X't) / 2M + 2k w / 2
= (X'X w - X't) / M + k w

Set dJ / dw = 0

We have
(X'X w - X't) / M + k w = 0

Then, w = (X'X + kMI)^(-1) X't
Here, I is the identity matrix.
Note: without the identity matrix, the solution is wrong.

c) increase lambda

d) Not a contradiction. The Gauss-Markov theorem assumes the true function is indeed linear. The bias-variance tradeoff assumes there is a true function f(x), which may not be linear. In reality, f(x) may not be linear, so we need the bias-variance tradeoff.

Some solutions said the Gauss-Markov theorem says OLS is the least variance only among unbiased estimates. But L2-penalized MSE is a biased estimate. This is not what I intended for, but I also gave full marks after I thought for a few times. I revised all the markings on this question. But if you still see mark deduction, please let me know.

Problem 5 [30 marks]

Suppose we are to minimize a loss function $J(\mathbf{w})$ where $\mathbf{w}$ is the parameters.

A Newton's update has the form $\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1}\mathbf{g}$
where $\mathbf{H} = \nabla^2_{\mathbf{w}} J(\mathbf{w})$ and $\mathbf{g} = \nabla_{\mathbf{w}} J(\mathbf{w})$, both evaluated at $\mathbf{w}^{(\text{old})}$. Here, we assume the loss is twice-differentiable and the Hessian is invertible.

We consider mean square error for linear regression $J(\mathbf{w}) = \frac{1}{2M}\|\mathbf{X}\mathbf{w} - \mathbf{t}\|^2$, where $\mathbf{X} \in \mathbb{R}^{M\times(d+1)}$ is the design matrix, $\mathbf{w} \in \mathbb{R}^{(d+1)}$ is the weights, and $\mathbf{t} \in \mathbb{R}^M$ is the target values of all data samples.

a) [10 marks] Suppose an initial weight is $\mathbf{w}_0$. Compute the weight after one step of Newton's update. Please provide some derivation steps.

b) [10 marks] Compare the above result with the closed-form solution of MSE, and explain why this happens.

Suppose J is a quadratic function of $\mathbf{w}$. Here, we do not assume we deal with the linear regression problem.

c) [10 marks] We now modify the original Newton's update as

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - a\mathbf{H}^{-1}\mathbf{g}$$

where a can be thought of as a learning rate.

Prove either in texts or equations that, if $0 < a < 1$, then $J(\mathbf{w}^{(\text{new})}) \leq J(\mathbf{w}^{(\text{old})})$.

Hints:

Useful matrix calculus identities

- If $\mathbf{A}$ is not a function of $\mathbf{x}$, then $\nabla_{\mathbf{x}}\mathbf{A}\mathbf{x} = \mathbf{A}^\top$
- If $\mathbf{A}$ is not a function of $\mathbf{x}$ and $\mathbf{A}$ is symmetric, then $\nabla_{\mathbf{x}}\mathbf{x}^\top\mathbf{A}\mathbf{x} = 2\mathbf{A}\mathbf{x}$

Notational suggestions

You may invent crappy notations, for example, w0 for $\mathbf{w}_0$, wo for $\mathbf{w}^{(\text{old})}$, and wn for $\mathbf{w}^{(\text{new})}$. Comment on your notations after equations.

Solution:

a)

J = (Xw - t)' (Xw - t) / 2M

= (w' X' X w - 2 t' X w + t't)/2M

g = [X'X w0 - X't] / M

H = X'X / M

After one step of Newton's update, we have

w1 = w0 - H^(-1)g = w0 - (X'X)^(-1) (X'X w0 - X't) = (X'X)^(-1) X't

b) The result is the same as the closed-form solution to MSE. The reason is that Newton's method assumes that the function is quadratic and directly optimizes to the global optimum by one step. Here, MSE is indeed a quadratic function, so one-step Newton's update gives exactly the closed-form solution

c) Assume J is quadratic. One update of Newton's method gives the globally optimum.

We know if 0 < a < 1, the w_new is in the middle of w* and wold, where w* is the global optimum. We know J(w*) < J(wold) because J(w*) is the optimality. So the weighted average of J(w*) and J(wold) is less than J(wold).