CS 274A Homework 4

Probabilistic Learning: Theory and Algorithms, CS 274A, Winter 2026

Due: 12 noon, Tuesday March 3rd, submit via Gradescope

Instructions and Guidelines for Homeworks

- Please answer all of the questions and submit your solutions to Gradescope (either hand-written or typed are fine as long as the writing is legible).
- All problems are worth equal points (10 points) unless otherwise stated. All homeworks will get equal weight in computation of the final grade for the class (with lowest-scoring homework being dropped).
- Please be sure to read the academic integrity policy on the course Web page, including in particular the policies on restrictions for use of generative AI for homework assignments.
- The homeworks are intended to help you better understand the concepts we discuss in class. It is important that you solve the problems yourself to help you learn and reinforce the material from class. If you don't do the homeworks you may have difficulty in the exams later in the quarter.
- In problems that ask you to derive or prove a result you should submit a complete mathematical proof (i.e., each line must follow logically from the preceding one, without "hand-waving"). Be as clear as possible in explaining your notation and in stating your reasoning as you go from line to line.
- If you can't solve a problem, you can discuss the high-level concepts *verbally* with another student (e.g., what concepts from the lectures or notes or text are relevant to a problem). However, you should not discuss any of the details of a solution with another student. In particular, do not look at (or show to any other student) *any written material* directly related to the homeworks, including other students' solutions or drafts of solutions, solutions from previous versions of this class, etc. The work you submit should be your own original work.
- If you need to you can look up standard results/definition/identities from textbooks, class notes, textbooks, other reference material (e.g., from the Web). If you base any part of your solution on material that we did not discuss in class, or is not in the class notes, or is not a standard known result, then you should provide a reference in terms of where the result is from, e.g., "based on material in Section 2.2 in ....." or a URL (e.g., Wikipedia).
- Please read each problem carefully. If you believe there is a typo, or some information is missing, or the problem is unclear, please post a question on the Ed discussion board.

Required Reading for Homework 4:

1. For Problems 1–5: Class Notes on Regression (Noteset 5 on class Website), Sections 1 to 5 and 7 and 8 (you can skip Section 6 if you wish).
2. For Problem 6: Class Notes on Classification (NoteSet 6 on the class Website)

Also be aware that may need to look up certain terms (e.g., in a textbook or Wikipedia), such as *convexity* or *positive semi-definite* if you don't recall what these terms are or have not seen them before.

Optional Reading for Homework 4: The following reading, from the Mathematics for Machine Learning (MML) text, `https://mml-book.github.io/book/mml-book.pdf`, also provide additional helpful background, if you have time:

1. Chapter 5.1 to 5.5 on Vector Calculus (e.g., derivatives of vectors and matrices)
2. Chapter 7 on Continuous Optimization
   (a) Section 7.1, pages 225-233
   (b) Section 7.3 up to start of 7.3.1, pages 236-239
3. Chapter 8, sections 8.1 and 8.2
4. Chapter 9 on Linear Regression, Sections 9.1 and 9.2

Page numbers above refer to the current online version as of February 2026. Feel free to optionally read the other sections of the Chapters above. The notation will differ in places to the notation we are using in class, but will be broadly similar.

Problem 1: Normal Equations for Least Squares (MSE) Regression

Assume we have training data in the form $D = \{(\underline{x}_i, y_i)\}, i = 1, \ldots, N$, where each $\underline{x}_i$ is a d-dimensional real-valued vector (with one component set to the constant 1 to allow for an intercept term) and where each y_i is a real-valued scalar. Assume we wish to fit a linear model of the form $\underline{\theta}^T \underline{x}$ where $\underline{\theta}$ is a d-dimensional parameter vector, where by "fit" we mean here that we want to find $\hat{\underline{\theta}}$ that minimizes $MSE(\underline{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \big(y_i - \underline{\theta}^T \underline{x}_i\big)^2$.

1. Prove that the solution to this problem can be written as the solution of a system of d linear equations (often referred to as the "normal equations") that can be written in the form $\mathbf{A}\underline{\theta} = \underline{b}$ where $\underline{\theta}$ has dimension $d \times 1$, $\mathbf{A}$ is a $d \times d$ matrix, and $\underline{b}$ is a $d \times 1$ vector. Starting from the definition of $MSE(\underline{\theta})$ above, carefully write out all steps in your proof, and clearly show how $\mathbf{A}$ and $\underline{b}$ are defined. If you need to assume as part of your solution that a particular matrix is full rank then assume so and state that you have assumed this.

2. Define the time complexity of minimizing $MSE(\underline{\theta})$ using the normal equations, given a dataset $D = \{(\underline{x}_i, y_i)\}, i = 1, \ldots, N$. Time complexity is defined as being "on the order of" (i.e., "big O") of some function of d and N, e.g., computing a sum of N terms has time complexity $O(N)$, multiplying a $1 \times N$ vector by a $N \times d$ matrix has time complexity $O(Nd)$, etc.

Problem 2: Computational Complexity for Fitting Linear Models using MSE

Consider the optimization problem in Problem 2: fitting a linear model by minimizing MSE, with d parameters and a d-dimensional input $\underline{x}$, with N data points. Answer the following questions below:

1. Assume we are using gradient descent algorithm (Section 7.1 in MML) to solve this problem.
   (a) Derive the gradient update equation for this problem
   (b) Define the time complexity of doing one gradient update (using all N data points) as a function of d and N.

2. Assume now that instead of the full gradient method, we use instead the stochastic gradient method (see Section 7.1.3 in MML) for this problem, where we use M randomly selected datapoints as the mini-batch size for each stochastic gradient update. Define the time complexity of doing one such stochastic gradient update, as a function of d, M, N. You can assume the order of datapoints is already randomized, i.e., no time needs to be spent selecting M random examples.

3. In the context of this problem (i.e., linear model, MSE loss) provide (a) one significant strength and (b) one significant weakness of each of the following optimization methods, relative to at least one of the other methods:
   (a) Normal equations
   (b) Gradient descent

(c) Stochastic gradient descent

Your strengths and weaknesses can comment on the relative computational complexity and numerical stability of each method, for example as a function of d relative to fixed N and fixed M; or could comment on whether a method might be sensitive to hyperparameters. For the iterative methods you can treat the number of iterations as some unknown constant for each method.

Problem 3: Estimating a Linear Model using Maximum Likelihood

Assume we have IID training data in the form $D = \{(x_i, y_i)\}, i = 1, \ldots, N$, where x_i and y_i are both one-dimensional and real-valued. Assume that y given x is a conditional Gaussian density with mean $E[y|x] = ax + b$ and with variance σ^2 (see example 8.4 in the MML text). Assume that a, b, and σ^2 are unknown.

Show from first principles that the maximum likelihood estimates for each of a, b, and σ^2 can be written as:

$$\hat{a} = \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - (\overline{x})^2}$$

$$\hat{b} = \overline{y} - \hat{a}\overline{x}$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_i \big(y_i - [\hat{a}x_i + \hat{b}]\big)^2$$

where terms such as $\bar{x}, \bar{y}$ represent empirical averages over the N datapoints, and terms like $\hat{a}$ represent maximum likelihood estimates.

Note that what is referred to likelihood above (and in the remaining problems in this homework) is actually the *conditional likelihood*, $P(D_y|D_x, \underline{\theta})$: see Sections 7 and 8 of NoteSet 5 on Regression.

Problem 4: Poisson Regression

Consider a problem where we have a data set $D = \{(\underline{x}_i, y_i)\}, i = 1, \ldots, N$ where $\underline{x}_i$ are real-valued d-dimensional vectors and $y_i \in \{0, 1, 2, \ldots, \}$, i.e., the y_i's are non-negative integers. For example, the y_i could be a count of the number of purchases that individual i makes in an online store per visit, and $\underline{x}_i$ could be a vector of features (e.g., demographic) for the individual. In a Poisson regression model we build a model where the conditional distribution of y, $P(y|\underline{x}; \underline{\theta})$, is assumed to be a Poisson distribution with mean $E[y|\underline{x}] = \lambda(\underline{x}) = f(\underline{x}; \underline{\theta})$ where the mean varies as a function of $\underline{x}$, for some fixed value of parameters $\underline{\theta}$, rather than being having a fixed mean value λ. To ensure that $\lambda(\underline{x}) > 0$, a common parametrization is $\lambda(\underline{x}) = exp(\underline{\theta}^T \underline{x})$, which is what we will use in this problem.

1. Derive the log-likelihood for this problem
2. Derive an equation for the gradient of the log-likelihood with respect to $\underline{\theta}$ for this problem

Problem 5: L1 or Lasso Regression

Consider a squared error loss function $MSE(\underline{\theta}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\underline{x}_i;\underline{\theta})^2$ with training data $D = \{(\underline{x}_i, y_i)\}, i = 1,\ldots,N$ and where f is some prediction model with unknown parameters $\underline{\theta} = (\underline{\theta}_1,\ldots,\underline{\theta}_p)$. A popular regularization method takes the form $r(\underline{\theta}) = \sum_{j=1}^{p}|\underline{\theta}_j|$, resulting in an optimization problem where we minimize $MSE(\underline{\theta}) + \lambda r(\underline{\theta})$ as a function of $\underline{\theta}$, rather than just minimizing $MSE(\underline{\theta})$. Here λ is the relative weight of the regularization term (this is known as L1 or Lasso regularization).

Clearly show how we can interpret L1 regularization in terms of a prior on $\underline{\theta}$ (by viewing this optimization problem from a Bayesian MAP perspective). Be sure to state clearly what distributional form this prior is, i.e., what name it has. (Note that you can assume here that the likelihood is Gaussian where the mean of y as a function of $\underline{x}$ is modeled by $f(\underline{x};\underline{\theta})$ and the variance term in the likelihood is known and fixed, as in Section 7 in NoteSet 5).

Problem 6: Convexity for Logistic Classifiers

Consider a classification problem where we have training data $D = \{(\underline{x}_i, y_i)\}, 1\ldots,i,\ldots,N$ where $\underline{x}_i$ are real-valued d-dimensional vectors and $y_i \in \{0,1\}$ are binary class labels. We will assume for convenience that the first component of $\underline{x}$ always takes value 1, allowing us to have an intercept (or bias) term in our model. Let $f(\underline{x};\underline{\theta})$ be a logistic regression model, where $\underline{\theta}$ is a d-dimensional parameter vector (set of weights), and our predictive model is

$$f(\underline{x};\underline{\theta}) \;=\; \frac{1}{1+\exp(-\underline{\theta}^T\underline{x})}.$$

where $f(\underline{x};\underline{\theta})$ is our estimate of $p(y=1|\underline{x})$.

Let the objective function (that we want to minimize) be the cross-entropy loss (also known as the log-loss), defined as

$$CE(\underline{\theta}) \;=\; -\frac{1}{N}\sum_{i=1}^{N} y_i \log f(x_i;\underline{\theta}) \;+\; (1-y_i)\log\big(1-f(x_i;\underline{\theta})\big).$$

1. Derive the equation for the gradient $\nabla_{\underline{\theta}}$ of $CE(\underline{\theta})$ for this optimization problem
2. Prove that $CE(\underline{\theta})$ has a single global minimum and no local minima by proving that it is a convex function of $\underline{\theta}$. Note that one way to prove this is to use the fact that a sufficient condition for a function like $CE(\underline{\theta})$ to be convex is that the Hessian matrix $\mathbf{H}_{\underline{\theta}}$ is positive semi-definite, where the Hessian matrix is the matrix of second partial derivatives of $CE(\underline{\theta})$ with respect to its parameters. There are also other ways to also prove convexity.
3. Iterative optimization methods for learning the parameters of a logistic model include first-order methods (which only use the gradient) and second-order methods (which use the Hessian). Define the computational (time) complexity as a function of d and N for computing (a) a gradient vector $\nabla_{\underline{\theta}}$, and (b) for computing a Hessian matrix $\mathbf{H}_{\underline{\theta}}$ and explain how you arrived at your solution.