

CS 229, Fall 2018

Problem Set #1: Supervised Learning

Due Wednesday, Oct 17 at 11:59 pm on Gradescope.

Notes: (1) These questions require thought, but do not require long answers. Please be as concise as possible. (2) If you have a question about this homework, we encourage you to post your question on our Piazza forum, at <http://piazza.com/stanford/fall2018/cs229>. (3) If you missed the first lecture or are unfamiliar with the collaboration or honor code policy, please read the policy on Handout #1 (available from the course website) before starting work. (4) For the coding problems, you may not use any libraries except those defined in the provided `environment.yml` file. In particular, ML-specific libraries such as scikit-learn are not permitted. (5) To account for late days, the due date listed on Gradescope is Oct 20 at 11:59 pm. If you submit after Oct 17, you will begin consuming your late days. If you wish to submit on time, submit before Oct 17 at 11:59 pm.

All students must submit an electronic PDF version of the written questions. We highly recommend typesetting your solutions via \LaTeX . If you are scanning your document by cell phone, please check the Piazza forum for recommended scanning apps and best practices. All students must also submit a zip file of their source code to Gradescope, which should be created using the `make.zip.py` script. In order to pass the auto-grader tests, you should make sure to (1) restrict yourself to only using libraries included in the `environment.yml` file, and (2) make sure your code runs without errors using the `run.py` script. Your submission will be evaluated by the auto-grader using a private test set.

Honor code: We strongly encourage students to form study groups. Students may discuss and work on homework problems in groups. However, each student must write down the solutions independently, and without referring to written notes from the joint session. In other words, each student must understand the solution well enough in order to reconstruct it by him/herself. In addition, each student should write on the problem set the set of people with whom s/he collaborated. Further, because we occasionally reuse problem set questions from previous years, we expect students not to copy, refer to, or look at the solutions in preparing their answers. It is an honor code violation to intentionally refer to a previous year's solutions.

1. [40 points] Linear Classifiers (logistic regression and GDA)

In this problem, we cover two probabilistic linear classifiers we have covered in class so far. First, a discriminative linear classifier: logistic regression. Second, a generative linear classifier: Gaussian discriminant analysis (GDA). Both the algorithms find a linear decision boundary that separates the data into two classes, but make different assumptions. Our goal in this problem is to get a deeper understanding of the similarities and differences (and, strengths and weaknesses) of these two algorithms.

For this problem, we will consider two datasets, provided in the following files:

- i. `data/ds1_{train,valid}.csv`
- ii. `data/ds2_{train,valid}.csv`

Each file contains m examples, one example $(x^{(i)}, y^{(i)})$ per row. In particular, the i -th row contains columns $x_0^{(i)} \in \mathbb{R}$, $x_1^{(i)} \in \mathbb{R}$, and $y^{(i)} \in \{0, 1\}$. In the subproblems that follow, we will investigate using logistic regression and Gaussian discriminant analysis (GDA) to perform binary classification on these two datasets.

(a) [10 points] In lecture we saw the average empirical loss for logistic regression:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})),$$

where $y^{(i)} \in \{0, 1\}$, $h_{\theta}(x) = g(\theta^T x)$ and $g(z) = 1/(1 + e^{-z})$.

Find the Hessian H of this function, and show that for any vector z , it holds true that

$$z^T H z \geq 0.$$

Hint: You may want to start by showing that $\sum_i \sum_j z_i x_i x_j z_j = (x^T z)^2 \geq 0$. Recall also that $g'(z) = g(z)(1 - g(z))$.

Remark: This is one of the standard ways of showing that the matrix H is positive semi-definite, written “ $H \succeq 0$.” This implies that J is convex, and has no local minima other than the global one. If you have some other way of showing $H \succeq 0$, you’re also welcome to use your method instead of the one above.

(b) [5 points] **Coding problem.** Follow the instructions in `src/p01b_logreg.py` to train a logistic regression classifier using Newton’s Method. Starting with $\theta = \vec{0}$, run Newton’s Method until the updates to θ are small: Specifically, train until the first iteration k such that $\|\theta_k - \theta_{k-1}\|_1 < \epsilon$, where $\epsilon = 1 \times 10^{-5}$. Make sure to write your model’s predictions to the file specified in the code.

(c) [5 points] Recall that in GDA we model the joint distribution of (x, y) by the following equations:

$$\begin{aligned} p(y) &= \begin{cases} \phi & \text{if } y = 1 \\ 1 - \phi & \text{if } y = 0 \end{cases} \\ p(x|y=0) &= \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_0)^T \Sigma^{-1} (x - \mu_0)\right) \\ p(x|y=1) &= \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_1)^T \Sigma^{-1} (x - \mu_1)\right), \end{aligned}$$

where ϕ , μ_0 , μ_1 , and Σ are the parameters of our model.

Suppose we have already fit ϕ , μ_0 , μ_1 , and Σ , and now want to predict y given a new point x . To show that GDA results in a classifier that has a linear decision boundary, show the **posterior distribution** can be written as

$$p(y = 1 \mid x; \phi, \mu_0, \mu_1, \Sigma) = \frac{1}{1 + \exp(-(\theta^T x + \theta_0))},$$

where $\theta \in \mathbb{R}^n$ and $\theta_0 \in \mathbb{R}$ are appropriate functions of ϕ , Σ , μ_0 , and μ_1 .

- (d) [7 points] For this part of the problem only, you may assume n (the dimension of x) is 1, so that $\Sigma = [\sigma^2]$ is just a real number, and likewise the determinant of Σ is given by $|\Sigma| = \sigma^2$. Given the dataset, we claim that the maximum likelihood estimates of the parameters are given by

$$\begin{aligned}\phi &= \frac{1}{m} \sum_{i=1}^m 1\{y^{(i)} = 1\} \\ \mu_0 &= \frac{\sum_{i=1}^m 1\{y^{(i)} = 0\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 0\}} \\ \mu_1 &= \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}} \\ \Sigma &= \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T\end{aligned}$$

The log-likelihood of the data is

$$\begin{aligned}\ell(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^m p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\ &= \log \prod_{i=1}^m p(x^{(i)} \mid y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi).\end{aligned}$$

By **maximizing ℓ with respect to the four parameters**, prove that the maximum likelihood estimates of ϕ , μ_0 , μ_1 , and Σ are indeed as given in the formulas above. (You may assume that there is at least one positive and one negative example, so that the denominators in the definitions of μ_0 and μ_1 above are non-zero.)

- (e) [3 points] **Coding problem.** In `src/p01e_gda.py`, fill in the code to calculate ϕ , μ_0 , μ_1 , and Σ , use these parameters to derive θ , and use the resulting GDA model to make predictions on the validation set.
- (f) [5 points] For Dataset 1, create a plot of the training data with x_1 on the horizontal axis, and x_2 on the vertical axis. To visualize the two classes, use a different symbol for examples $x^{(i)}$ with $y^{(i)} = 0$ than for those with $y^{(i)} = 1$. On the same figure, plot the decision boundary found by logistic regression in part (b). Make an identical plot with the decision boundary found by GDA in part (e).
- (g) [5 points] Repeat the steps in part (f) for Dataset 2. On which dataset does GDA seem to perform worse than logistic regression? Why might this be the case?
- (h) [3 extra credit points] For the dataset where GDA performed worse in parts (f) and (g), can you find a transformation of the $x^{(i)}$'s such that GDA performs significantly better? What is this transformation?