CSCI-GA.2565 — Homework 2

your NetID here

Version 1.0

Instructions.

- **Due date.** Homework is due **Wednesday, February 26, at noon EST**.
- **Gradescope submission.** Everyone must submit individually at gradescope under `hw2` and `hw2code`: `hw2code` is just python code, whereas `hw2` contains everything else. For clarity, problem parts are annotated with where the corresponding submissions go.
  - **Submitting `hw2`.** `hw2` must be submitted as a single PDF file, and typeset in some way, for instance using LaTeX, Markdown, Google Docs, MS Word; you can even use an OCR package (or a modern multi-modal LLM) to convert handwriting to LaTeXand then clean it up for submission. Graders reserve the right to award zero points for solutions they consider illegible.
  - **Submitting `hw2code`.** Only upload the two python files `hw2.py` and `hw2_utils.py`; don't upload a zip file or additional files.
- **Consulting LLMs and friends.** You may discuss with your peers and you may use LLMs. *However,* you are strongly advised to make a serious attempt on all problems alone, and if you consult anyone, make a serious attempt to understand the solution alone afterwards. You must document credit assignment in a special final question in the homework.
- **Evaluation.** We reserve the right to give a 0 to a submission which violates the intent of the assignment and is morally equivalent to a blank response.
  - `hw2code:` your grade is what the autograder gives you; note that you may re-submit as many times as you like until the deadline. However, we may reduce your auto-graded score if your solution simply hard-codes answers.
  - `hw2:` you can receive 0 points for a blank solution, an illegible solution, or a solution which does not correctly mark problem parts with boxes in the gradescope interface (equivalent to illegibility). All other solutions receive full points, *however* the graders do leave feedback so please check afterwards even if you received a perfect score.
- **Regrades.** Use the grade scope interface.
- **Late days.** We track 3 late days across the semester per student.
- **Library routines.** Coding problems come with suggested "library routines"; we include these to reduce your time fishing around APIs, but you are free to use other APIs.

Version history.

1.0. Initial version.

1. This problem is about SVMs over $\mathbb{R}^d$ with linearly separable data (i.e., the hard margin SVM).

Our formulation of SVM required separators to pass through the origin, which does not provide a geometrically pleasing notion of maximum margin direction.

A first fix is provided by lecture 4: by appending a 1 to the inputs, we obtain the convex program

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & \frac{1}{2}\|\boldsymbol{u}\|^2 \\ \text{subject to} \quad & \boldsymbol{u} \in \mathbb{R}^{d+1} \\ & y_i \left[\begin{smallmatrix} \boldsymbol{x}_i \\ 1 \end{smallmatrix}\right]^{\mathsf{T}} \boldsymbol{u} \geq 1 \qquad \forall i, \end{aligned}$$

and let $\bar{\boldsymbol{u}}$ denote the optimal solution to this program.

A second standard fix is to incorporate the bias directly into the optimization problem:

$$\begin{aligned} \min_{\boldsymbol{v},b} \quad & \frac{1}{2}\|\boldsymbol{v}\|^2 \\ \text{subject to} \quad & \boldsymbol{v} \in \mathbb{R}^{d}, b \in \mathbb{R} \\ & y_i(\boldsymbol{v}^{\mathsf{T}}\boldsymbol{x}_i + b) \geq 1 \qquad \forall i, \end{aligned}$$

and let $(\bar{\boldsymbol{v}}, \bar{b}) \in \mathbb{R}^d \times \mathbb{R}$ denote an optimal solution to this program. This second version is standard, but we do not use it in lecture for various reasons.

(a) [hw2] In lecture, we stated that the first formulation is a *convex program* (formally defined in lecture 5). Show that the second formulation is also a convex program.

(b) [hw2] Suppose there is only one datapoint: $\boldsymbol{x}_1 = \boldsymbol{e}_1$, the first standard basis vector, with label $y_1 = +1$. The first formulation will have a unique solution $\bar{\boldsymbol{u}}$, as discussed in lecture. Show that the second formulation does not have a unique solution.

(c) [hw2] Let's add another datapoint: $\boldsymbol{x}_2 = -a\boldsymbol{e}_1$ for some $a \geq 3$, with label $y_2 = -1$. Now that we have two data points, both of the convex programs now have two constraints. Write out the explicit constraints to the first convex program.

(d) [hw2] Using these two constraints, show that the first coordinate $\bar{u}_1$ of the optimal solution $\bar{\boldsymbol{u}}$ satisfies $\bar{u}_1 \geq \frac{2}{a+1}$.

(e) [hw2] Using parts (c) and (d), find optimal solutions $\bar{\boldsymbol{u}}$ and $(\bar{\boldsymbol{v}}, \bar{b})$, and prove they are in fact optimal.

Hint: If you are stuck, first try the case $d = 1$. Then study what happens for $d = 2, d = 3, \ldots$

Hint: $(\bar{\boldsymbol{v}}, \bar{b})$ will be unique.

(f) [hw2] Now we will consider the behavior of $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{v}}$ as a increases; to this end, write $\bar{\boldsymbol{u}}_a$ and $\bar{\boldsymbol{v}}_a$, and consider $a \to \infty$. Determine and formally prove the limiting behavior of $\lim_{a\to\infty} \frac{1}{2}\|\bar{\boldsymbol{u}}_a\|^2$ and $\lim_{a\to\infty} \frac{1}{2}\|\bar{\boldsymbol{v}}_a\|^2$.

Hint: The two limits will not be equal.

(g) [hw2] Between the two versions of SVM with bias, which do you prefer? Any answer which contains at least one complete sentence will receive full credit.

Remark: Initially it may have seemed that both optimization problems have the same solutions; the purpose of this problem was to highlight that small differences in machine learning methods can lead to observably different performance.

Solution.

2. SVM Implementation.

Recall that the dual problem of an SVM is

$$\max_{\boldsymbol{\alpha}\in\mathcal{C}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

where the domain $\mathcal{C} = [0,\infty)^n = \{\boldsymbol{\alpha} : \alpha_i \geq 0\}$ for a hard-margin SVM, and $\mathcal{C} = [0, C]^n = \{\boldsymbol{\alpha} : 0 \leq \alpha_i \leq C\}$ for a soft-margin SVM. Equivalently, we can frame this as the minimization problem

$$\min_{\boldsymbol{\alpha}\in\mathcal{C}} f(\boldsymbol{\alpha}) := \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i=1}^{n} \alpha_i.$$

This can be solved by projected gradient descent, which starts from some $\boldsymbol{\alpha}_0 \in \mathcal{C}$ (e.g., $\mathbf{0}$) and updates via

$$\boldsymbol{\alpha}_{t+1} = \Pi_{\mathcal{C}}\left[\boldsymbol{\alpha}_t - \eta\nabla f(\boldsymbol{\alpha}_t)\right],$$

where $\Pi_{\mathcal{C}}[\boldsymbol{\alpha}]$ is the *projection* of $\boldsymbol{\alpha}$ onto $\mathcal{C}$, defined as the closest point to $\boldsymbol{\alpha}$ in $\mathcal{C}$:

$$\Pi_{\mathcal{C}}[\boldsymbol{\alpha}] := \arg\min_{\boldsymbol{\alpha}'\in\mathcal{C}} \|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_2.$$

If $\mathcal{C}$ is convex, the projection is uniquely defined.

(a) [`hw2`] Prove that

$$\left(\Pi_{[0,\infty)^n}[\boldsymbol{\alpha}]\right)_i = \max\{\alpha_i, 0\},$$

and

$$\left(\Pi_{[0,C]^n}[\boldsymbol{\alpha}]\right)_i = \min\{\max\{0, \alpha_i\}, C\}.$$

Hint: Show that the ith component of any other $\boldsymbol{\alpha}' \in \mathcal{C}$ is further from the ith component of $\boldsymbol{\alpha}$ than the ith component of the projection is. Specifically, show that $\left|\alpha_i' - \alpha_i\right| \geq \left|\max\{0, \alpha_i\} - \alpha_i\right|$ for $\boldsymbol{\alpha}' \in [0,\infty)^n$ and that $\left|\alpha_i' - \alpha_i\right| \geq \left|\min\left\{\max\{0, \alpha_i\}, C\right\} - \alpha_i\right|$ for $\boldsymbol{\alpha}' \in [0, C]^n$.

(b) [`hw2code`] Implement an `svm_solver()`, using projected gradient descent formulated as above. Initialize your $\boldsymbol{\alpha}$ to zeros. See the docstrings in `hw2.py` for details.

Remark: Consider using the `.backward()` function in pytorch. However, then you may have to use in-place operations like `clamp_()`, otherwise the gradient information is destroyed.

Library routines: `torch.outer, torch.clamp, torch.autograd.backward, torch.tensor(..., requires_grad=True), with torch.no_grad():, torch.tensor.grad.zero_, torch.tensor.detach.`

(c) [`hw2code`] Implement an `svm_predictor()`, using an optimal dual solution, the training set, and an input. See the docstrings in `hw2.py` for details.

Library routines: `torch.empty.`

(d) [`hw2`] On the area $[-5, 5] \times [-5, 5]$, plot the contour lines of the following kernel SVMs, trained on the XOR data. Different kernels and the XOR data are provided in `hw2_utils.py`. Learning rate 0.1 and 10000 steps should be enough. To draw the contour lines, you can use `hw2_utils.svm_contour()`.

- The polynomial kernel with degree 2.
- The RBF kernel with $\sigma = 1$.
- The RBF kernel with $\sigma = 2$.
- The RBF kernel with $\sigma = 4$.

Include these four plots in your written submission.

Solution.

3. Convexity.

In this problem, you will analyze a convex approximation of the max function and get familiar with techniques establishing convexity of a function. Denote the max function $\phi : \mathbb{R}^n \to \mathbb{R}$ and its approximation $\psi : \mathbb{R}^n \to \mathbb{R}$ as

$$\phi(x) := \max(x_1, \ldots, x_n), \quad \text{and} \quad \psi(x) := \ln\left(\sum_{i=1}^{n} \exp(x_i)\right).$$

Furthermore, throughout the problem, for any vector $x \in \mathbb{R}^n$, denote $\exp(x) := \big(\exp(x_1), \ldots, \exp(x_n)\big)$.

(a) [hw2] Prove that

$$\phi(x) \leq \psi(x) \leq \phi(x) + \ln(n).$$

Hint: Show that $\phi(\exp(x)) \leq \sum_{i=1}^{n} \exp(x_i) \leq n \cdot \phi(\exp(x))$.

Remark: Part (a) quantifies how well ψ approximates the max function.

(b) [hw2] Use part (a) to show that

$$\lim_{c \to \infty} \frac{\psi(cx)}{c} = \phi(x).$$

(c) [hw2] Prove that the max function ϕ is convex.

(d) [hw2] Compute the Hessian $\nabla^2 \psi$.

(e) [hw2] Define $\lambda_i := \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)}$ for $i \in [n]$. Rewrite the Hessian in part (d) in terms of $\{\lambda_1, \ldots, \lambda_n\}$.

(f) [hw2] Show that the Hessian $\nabla^2 \psi(x)$ is positive semi-definite for all $x \in \mathbb{R}^n$. From lecture 3, it follows that ψ is convex.

Hint: An equivalent definition of a positive semi-definite matrix $M \in \mathbb{R}^{n \times n}$ is that for any $v \in \mathbb{R}^n$, $v^\intercal M v \geq 0$. Use this definition, part (e), and Jensen's inequality (see appendix to lecture 3).

(g) [hw2] Directly show that for all $\alpha \in [0, 1]$ and for any $x, y \in \mathbb{R}^n$,

$$\psi\left(\alpha x + (1 - \alpha) y\right) \leq \alpha \psi(x) + (1 - \alpha) \psi(y).$$

Hint: Fix $x, y \in \mathbb{R}^n$ and denote $a = \exp(x)$ and $b = \exp(y)$. Write $\psi\left(\alpha x + (1 - \alpha) y\right)$ as $\ln\left(\sum_{i=1}^{n} a_i^{\alpha} b_i^{(1-\alpha)}\right)$ and apply Hölder's inequality (see appendix to lecture 3).

Remark: This gives an alternate proof that ψ is convex. Note that this proof does not use the fact that ψ is twice differentiable.

Solution.

4. LLM Use and Other Sources.

[hw2] Please document, in detail, all your sources, including include LLMs, friends, internet resources, etc. For example:

1a. I asked my friend, then I found a different way to derive the same solution.

1b. ChatGPT 4o solved the problem in one shot, but then I rewrote it once one paper, and a few days later tried to re-derive an answer from scratch.

1c. I accidentally found this via a google search, and had trouble forgetting the answer I found, but still typed it from scratch without copy-paste.

1d. ...

⋮

6. I used my solution to problem 5 to write this answer.

Solution.