TTIC 31020: Introduction to Machine Learning **Winter 2025**

Homework 7

Due: 3 PM, Feb 20th, 2025

Note *You may discuss these problems in groups. However, you must write up your own solutions and mention the names of the people in your group. Also, please do mention any books, papers or other sources you refer to. We prefer and encourage that you typeset solutions in LaTeX for written assignments. Handwritten solutions that are not neat, organized and with clear and easy-to-read handwriting will not be graded. Please submit your solutions as a PDF document on Canvas.*

Challenge and Optional Questions *Challenge questions are marked with* challenge *and optional questions are marked with* optional *. You can get extra credit for solving* challenge *questions. You are not required to turn in* optional *questions, but we encourage you to solve them for your understanding. Course staff will help with* optional *questions but will prioritize queries on non-*optional *questions.*

1. **Kernelizing Gradient Descent** In this question we will consider empirical risk minimization (ERM) for linear predictors, where recall the empirical risk can written as;

$$L_S(w) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle w, \phi(x_i)\rangle ; y_i) \tag{1}$$

Our goal will be to implement gradient descent on the empirical risk where instead of representing the iterates $w^{(t)}$ explicitly, we only represent them through coefficient $\alpha^{(t)}$ such that $w^{(t)} = \sum_{i=1}^{m} \alpha_i^{(t)} \phi(x_i)$, and instead of using $\phi(x)$ explicitly, we only access it through the Kernel function $K(x, x') = \langle \phi(x), \phi(x')\rangle$.

   (a) Consider the gradient descent updates:

$$w^{(0)} = 0, \qquad w^{(t+1)} = w^{(t)} - \eta \nabla_w L_S(w^{(t)}) \tag{2}$$

   Write down $\alpha^{(t+1)}$ in terms of $\alpha^{(t)}$, $K(x_i, x_j)$, y_i and $\ell'(\cdot,\cdot)$, without referring directly to $\phi(.)$ or $w^{(t)}$. Denoting T_K for the runtime of each kernel evaluation, write down the time complexity of each iteration of gradient descent when implemented this way (e.g. $O(T_k m \log m + m^3)$ represents $O(m \log m)$ kernel evaluation and $O(m^3)$ additional operations).

   (b) Turning to ℓ_2 Regularized Empirical Risk Minimization (RERM), where we minimize the regularized objective

$$F_2(w) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle w, \phi(x_i)\rangle ; y_i) + \frac{\lambda}{2}\|w\|_2^2 \tag{3}$$

   using gradient descent, i.e. updates $w^{(t+1)} = w^{(t)} - \eta \nabla_w F_2(w^{(t)})$, write down $\alpha^{(t+1)}$ in terms of $\alpha^{(t)}$, $K(x_i, x_j)$, y_i and $\ell'(\cdot,\cdot)$, without referring directly to $\phi(.)$ or $w^{(t)}$.

   (c) Now consider ℓ_1 Regularized Empirical Risk Minimization (RERM), where we minimize the regularized objective

$$F_1(w) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle w, \phi(x_i)\rangle ; y_i) + \frac{\lambda}{2}\|w\|_1 \tag{4}$$

   Is it possible to write down the iterates in terms of $K(x_i, x_j)$ without referring to $\phi(x)$ directly? Is it possible to express the gradient descent iterates on F_1 as $w^{(t)} = \sum_{i=1}^{m} \alpha_i^{(t)} \phi(x_i)$?
   (Hint: think of rotational invariance and of whether the iterates stay in the span of the data).

(d) An alternative approach is to use gradient descent w.r.t. α, that is updates of the form:

$$\alpha^{(t+1)} = \alpha^{(t)} - \eta \nabla_\alpha L_S \left(w\left(\alpha^{(t)}\right)\right). \tag{5}$$

where $w(\alpha) = \sum_i \alpha_i \phi(x_i)$. Write down the update (5) explicitly, again in terms of $K(x_i, x_j)$ rather than $\phi(x_i)$, and contrast it with the α-representation of the update (2) you found in question 1a. Using a simple numerical example (computed explicitly or on a computer) show that the two updates can be in a different directions.

challenge Can you write down a simple relationship between the two updates, i.e. express one update direction in terms of the other?

optional challenge Which update do you think would be better? Test their convergence speeds empirically.

(e) We now want to consider the *Stochastic* Gradient Descent (SGD) update, based only on a single training example $x_{i^{(t)}}, y_{i^{(t)}}$, and given by:

$$w^{(0)} = 0, \qquad w^{(t+1)} = w^{(t)} - \eta \nabla_w \ell(\left\langle w^{(t)}, \phi(x_{i^{(t)}})\right\rangle; y_{i^{(t)}}). \tag{6}$$

Write down $\alpha^{(t+1)}$ in terms of $i^{(t)}$,$\alpha^{(t)}$, $K(x_i, x_j)$, y_i and $\ell'(\cdot,\cdot)$ corresponding to the update (6). How many coefficients α_i are updated at each iteration? What is the time complexity of each iteration, again using T_K to denote the kernel evaluation time? Note that unlike unkernelized SGD, where the runtime per iteration is independent of m, here even the SGD runtime scales with m.

(f) optional Now consider SGD on the regularized objective F_2, where again in each iteration we use a stochastic gradient estimate based on a single training point. Write down the update for $\alpha^{(t+1)}$ as above. How many coefficients are being updated?

optional challenge Suggest a modified representation that allows implementing the Kernalized SGD update in time $O(mT_K)$, and where only two coefficients are updated at each iteration (Hint: suggest an overparametrization that uses one additional, redundant, coefficient).

2. **Implicit Regularization in Gradient Descent**

Consider the dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ and a feature map $\phi : \mathcal{X} \to \mathbb{R}^d$ such that $d > m$.

We consider the least squares linear regression problem that you already saw in Homework 6. Let $\Phi \in \mathbb{R}^{m\times d}$ whose i^{th} row is $\phi(x_i)$ and $y \in \mathbb{R}^m$ whose entries are the labels y_i s. For simplicity, assume that Φ has full row-rank, i.e. $\text{Rank}(\Phi) = m$.[1] Consider the following training objective

$$L_S(w) = \frac{1}{m}\|\Phi w - y\|^2.$$

Note that, when $d > m$, the objective $L_S(w)$ has multiple global minimizers w such that $L_S(w) = 0$.

Gradient Descent. Consider the gradient descent update rule that you already saw in Homework 6.

$$w^{(0)} = 0, \qquad w^{(t+1)} = w^{(t)} - \eta \nabla_w L_S(w^{(t)}).$$

(a) Show that there is a unique w^* from $\text{Span}\{\phi(x_1), \ldots, \phi(x_m)\}$ such that $L_S(w^*) = 0$. Moreover, it is given by $w^* := \Phi^\top(\Phi\Phi^\top)^{-1}y$. Argue that w^* must also the unique minimum ℓ_2 norm zero training error solution from $\mathbb{R}^d$. In other words, show that

$$w^* = \arg \min_{w\in\mathbb{R}^d, L_S(w)=0} \|w\|_2.$$

(b) Show, by induction on t, that $w^{(t)}$ is always in $\text{Span}\{\phi(x_1), \ldots, \phi(x_m)\}$. Moreover, since $L_S(w)$ is convex in w, for a tuned step-size η, the algorithm converges to a global minimizer w^{GD} such that $L_S(w^{\text{GD}}) = 0$. Conclude that $w^{GD} = w^*$ and hence the gradient descent converges to the minimum ℓ_2 norm interpolating solution.

[1] If rows are linearly dependent, then remove some rows (i.e. examples) such that the resultant Φ has only independent rows.

3. **Non-separable Perceptron and Online-to-Batch Strikes Back** challenge Recall the Perceptron algorithm that you already analyzed in Homework 4 (Problem 1) and the entire setup therein! You already showed a mistake bound of $O(1/\gamma^2)$, assuming the data (bounded within radius 1) is linearly separable with a margin γ by some w^*. In this question, we shall see how to analyze the Perceptron algorithm, even when the data may not be linearly separable. Instead of assuming the data is exactly separable with a margin, derive a mistake bound in terms of the best possible total hinge loss on the sequence. For an instance sequence $(x_t, y_t)_{t=1}^m$, where $\phi(x_t) \in \mathbb{R}^d$, $y_t \in \{\pm 1\}$, and any $w \in \mathbb{R}^d$, define its total hinge loss as:

$$H(w) = \sum_{t=1}^{m} [1 - y_t \langle w, \phi(x_t) \rangle]_+$$

(a) Prove that for any sequence, with $\|\phi(x_t)\| \leq 1$, and any w^*, the number of mistakes M_m made by the (standard) Perceptron rule is bounded by:

$$M_m \leq H(w^*) + \|w^*\|^2 + \|w^*\| \sqrt{H(w^*)}$$

Hint: Follow the separable perceptron analysis, bounding $\|w_{t+1}\|^2$ from above in terms of M_t, and $\langle w^*, w_{t+1} \rangle$ from below in terms of both M_t and $H_t^* = \sum_{i=1}^{t} [1 - y_i \langle w^*, \phi(x_i) \rangle]_+$.

(b) Use this to design and analyze a batch learning rule A, such that for any source distribution $(X, Y) \sim \mathcal{D}$ over $\mathbb{R}^d \times \{\pm 1\}$ with $\mathbb{P}[\|\phi(X)\| \leq 1] = 1$, and any $w^* \in \mathbb{R}^d$ with expected hinge loss $L_{\text{hinge}}(w^*) = \mathbb{E}[1 - Y \langle w^*, X \rangle]_+$:

$$\mathbb{E}_{S \sim \mathcal{D}^m}[L_{\mathcal{D}}(A(S))] \leq L_{\text{hinge}}(w^*) + O\left(\frac{\|w^*\|^2}{m} + \sqrt{\frac{\|w^*\|^2 L_{\text{hinge}}(w^*)}{m}} \right)$$

where $L_{\mathcal{D}}(h) = \mathbb{E}[\mathbf{1}_{h(X) \neq Y}]$ is the standard misclassification error rate.