CS489/698: Introduction to Machine Learning

Homework 3

Due: 11:59 pm, October 31, 2017, submit on LEARN.

Include your name, student number and session!

Submit your writeup in pdf and all source code in a zip file (with proper documentation). Write a script for each programming exercise so that the TAs can easily run and verify your results. Make sure your code runs! [Text in square brackets are hints that can be ignored.]

Exercise 1: (Kernel) Logistic Regression (40 pts)

Note: In the interest of time, for this exercise you do not have to use cross-validation to select hyperparameters. Simply try a few (even a single one will receive full credit) and report the error for each hyperparameter. In case of memory issues, it is OK to use only the first training batch (10k images) in the CIFAR-10 dataset for training.

Algorithm 1: binary logistic regression.

Input: $X \in \mathbb{R}^{n\times d}$, $\mathbf{y} \in \{-1,1\}^n$ (training set), regularization constant $\lambda \geq 0$, initializer $\mathbf{w} \in \mathbb{R}^d$
Output: $\mathbf{w} \in \mathbb{R}^d$

repeat
  $\mathbf{w} \leftarrow \mathbf{w} - [\nabla^2 f(\mathbf{w})]^{-1} \cdot \nabla f(\mathbf{w})$ `// solve linear system instead of inverting the Hessian!`
until *convergence*

Consider the following binary logistic regression problem:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{\frac{1}{n_+}\sum_{i:y_i=1}\log(1+\exp(-y_i\mathbf{w}^\top\mathbf{x}_i)) + \frac{1}{n_-}\sum_{j:y_j=-1}\log(1+\exp(-y_j\mathbf{w}^\top\mathbf{x}_j)) + \lambda\|\mathbf{w}\|_2^2}_{f(\mathbf{w})}, \tag{1}$$

where $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times d}$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)^\top \in \{-1,1\}^n$ are the training data, n_+ is the number of training samples in the positive class and n_- is the number of training samples in the negative class, and $\lambda \geq 0$ is the regularization parameter. We introduce the two normalization constants n_+ and n_- in (1) in case the two classes are imbalanced (i.e., one class "overwhelms" the other).

1. (10 pts) Implement the Newton algorithm in Algorithm 1 for solving (1). Please include the derivation of the gradient ∇f and the Hessian $\nabla^2 f$ in your writeup. [For simplicity we do not include the bias term here, since we can append the constant 1 in each feature vector $\mathbf{x}_i$.] [To stop the algorithm, set a maximum number of iterations but also quit the iteration if the norm of the successive difference between the previous $\mathbf{w}$ and the current $\mathbf{w}$ falls below some relative tolerance say 1e-4.]

   Ans: Let $\tilde{y}_i = \frac{y_i+1}{2}$. Since $y_i \in \{1,-1\}$ we have $\tilde{y}_i \in \{0,1\}$. As is usual, let $p_i = \frac{1}{1+\exp(-\mathbf{w}^\top\mathbf{x}_i)}$. For the first term in (1), its derivative is:

$$\frac{1}{n_+}\sum_{i:y_i=1}\frac{\exp(-\mathbf{w}^\top\mathbf{x}_i)}{1+\exp(-\mathbf{w}^\top\mathbf{x}_i)}(-\mathbf{x}_i) = \sum_{i:y_i=1}\frac{p_i-1}{n_+}\mathbf{x}_i = \sum_{i:y_i=1}\frac{p_i-\tilde{y}_i}{\tilde{y}_i n_+ + (1-\tilde{y}_i)n_-}\mathbf{x}_i. \tag{2}$$

   Similarly, the second term in (1) has derivative:

$$\frac{1}{n_-}\sum_{j:y_j=-1}\frac{\exp(\mathbf{w}^\top\mathbf{x}_j)}{1+\exp(\mathbf{w}^\top\mathbf{x}_j)}\mathbf{x}_j = \sum_{j:y_j=-1}\frac{p_j}{n_-}\mathbf{x}_j = \sum_{j:y_j=-1}\frac{p_j-\tilde{y}_j}{\tilde{y}_j n_+ + (1-\tilde{y}_j)n_-}\mathbf{x}_j. \tag{3}$$

   Thus, the total derivative of f in (1) is:

$$\nabla f(\mathbf{w}) = 2\lambda\mathbf{w} + \sum_{i=1}^{n}\frac{p_i-\tilde{y}_i}{\tilde{y}_i n_+ + (1-\tilde{y}_i)n_-}\mathbf{x}_i. \tag{4}$$

Taking derivative of the above again (only p_i depends on $\mathbf{w}$), we have:

$$\nabla^2 f(\mathbf{w}) = 2\lambda \mathbb{I}_d + \sum_{i=1}^{n} \frac{p_i(1-p_i)}{\tilde{y}_i n_+ + (1-\tilde{y}_i)n_-} \mathbf{x}_i \mathbf{x}_i^\top, \tag{5}$$

where we have used the fact that:

$$\frac{\mathrm{d}p_i}{\mathrm{d}\mathbf{w}} = \frac{\exp(-\mathbf{w}^\top \mathbf{x}_i)}{(1+\exp(-\mathbf{w}^\top \mathbf{x}_i))^2} \mathbf{x}_i = \frac{1+\exp(-\mathbf{w}^\top \mathbf{x}_i)-1}{(1+\exp(-\mathbf{w}^\top \mathbf{x}_i))^2} \mathbf{x}_i = p_i(1-p_i)\mathbf{x}_i. \tag{6}$$

2. (10 pts) Test your algorithm on the CIFAR-10 dataset, using the one-vs-all strategy. Note that there are 10 classes in CIFAR-10 while (1) is for binary classes. The one-vs-all strategy trains 10 classifiers, each time with some c-th class as positive and the rest as negative. For prediction, run all 10 classifiers and predict with the classifier that has the highest dot product (probability/confidence). Report the test error and the corresponding value of λ that you tried. [To save time, you may skip cross-validation on λ but try a few λ's and report the test error of each.]

3. (10 pts) Test your algorithm on the CIFAR-10 dataset, using the one-vs-one strategy. Note that there are 10 classes in CIFAR-10 while (1) is for binary classes. The one-vs-one strategy trains 45 classifiers, each time with some c-th class as positive and some c'-th class as negative. For prediction, run all 45 classifiers, each of which will vote either for some class c or some class c'. Predict with the class that has most votes (break ties arbitrarily). Report the test error and the corresponding value of λ that you tried. [To save time, you may skip cross-validation on λ but try a few λ's and report the test error of each.]

4. (10 pts) In this exercise we take only the positive class "dog" and the negative class "cat" from the CIFAR-10 dataset (available on course website). Implement the **kernel** binary logistic regression and include the derivation of your algorithm in the writeup (including the gradient and the Hessian). Experiment with the linear kernel $\kappa_\ell(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$, the inhomogeneous polynomial kernel $\kappa_p(\mathbf{x}, \mathbf{x}') = (1+\mathbf{x}^\top \mathbf{x}')^5$ and the Gaussian kernel $\kappa_g(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x}-\mathbf{x}'\|_2^2/\sigma)$. Report the test error for each kernel and regularization constant λ (and kernel parameter σ). [To save time, you may skip cross-validation but report the test error for each configuration of parameters that you try.]

   Ans: Using the representer theorem, we know at optimality

$$\mathbf{w} = \sum_i \alpha_i \phi(\mathbf{x}_i). \tag{7}$$

   (Note that we have chosen to absorb y_i into α_i in the above. Of course, you can also take $\mathbf{w} = \sum_i \alpha_i y_i \phi(\mathbf{x}_i)$, and adapt the following derivations slightly.) Plugging back to (1) we have:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{1}{n_+} \sum_{i:y_i=1} \log(1+\exp(-y_i \boldsymbol{\alpha}^\top K_{:i})) + \frac{1}{n_-} \sum_{j:y_j=-1} \log(1+\exp(-y_j \boldsymbol{\alpha}^\top K_{:j})) + \lambda \boldsymbol{\alpha}^\top K \boldsymbol{\alpha}. \tag{8}$$

   From the apparent similarity with (1) (upon identifying $\mathbf{x}_i$ with $K_{:i}$), we immediately have:

$$\nabla f(\boldsymbol{\alpha}) = 2\lambda K \boldsymbol{\alpha} + \sum_{i=1}^{n} \frac{p_i - \tilde{y}_i}{\tilde{y}_i n_+ + (1-\tilde{y}_i)n_-} K_{:i}, \tag{9}$$

   where $\tilde{y}_i = \frac{1+y_i}{2}$ is as before while now

$$p_i = \frac{1}{1+\exp(-\boldsymbol{\alpha}^\top K_{:i})}. \tag{10}$$

Similarly, we have

$$\nabla^2 f(\mathbf{w}) = 2\lambda K + \sum_{i=1}^{n} \frac{p_i(1-p_i)}{\tilde{y}_i n_+ + (1-\tilde{y}_i)n_-} K_{:i} K_{:i}^\top, \tag{11}$$

Exercise 2: Kernel least squares (45 pts)

Convention: We use $\mathbb{I}_p$ to denote the $p \times p$ identity matrix.

Consider ridge regression that we discussed in lecture 02:

$$\min_{\mathbf{w}} \ \|X\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_2^2, \tag{12}$$

where $X \in \mathbb{R}^{n\times d}, \mathbf{w} \in \mathbb{R}^d, \mathbf{y} \in \mathbb{R}^n$ and $\lambda > 0$ is a tuning hyper-parameter. Recall the following closed-form optimal solution:

$$\mathbf{w}^\star = (X^\top X + \lambda \mathbb{I}_d)^{-1} X^\top \mathbf{y}. \tag{13}$$

In this exercise you are going to derive kernelized ridge regression.

1. (15 pts) Derive the Lagrangian dual of (12) by introducing a "dummy constraint":

$$\min_{\mathbf{w},\mathbf{z}} \ \|\mathbf{z}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \tag{14}$$

$$\text{s.t. } \mathbf{z} = X\mathbf{w} - \mathbf{y}. \tag{15}$$

Solve the Lagrangian dual and get a "different" closed-form solution for $\mathbf{w}$:

$$\mathbf{w}^\star = X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1} \mathbf{y}. \tag{16}$$

Ans: Introducing a Lagrangian multiplier $\boldsymbol{\alpha}$ for the "dummy constraint," we can get the Lagrangian:

$$\min_{\mathbf{w},\mathbf{z}} \max_{\boldsymbol{\alpha}} \|\mathbf{z}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 + \boldsymbol{\alpha}^\top (\mathbf{z} - X\mathbf{w} + \mathbf{y}) = \max_{\boldsymbol{\alpha}} \min_{\mathbf{w},\mathbf{z}} \|\mathbf{z}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 + \boldsymbol{\alpha}^\top (\mathbf{z} - X\mathbf{w} + \mathbf{y}). \tag{17}$$

Take derivative w.r.t. $\mathbf{w}$ and set it to zero:

$$2\lambda \mathbf{w} - X^\top \boldsymbol{\alpha} = \mathbf{0}. \tag{18}$$

Take derivative w.r.t. $\mathbf{z}$ and set it to zero:

$$2\mathbf{z} + \boldsymbol{\alpha} = \mathbf{0}. \tag{19}$$

Plug (18) and (19) back to (17) and simplify:

$$\max_{\boldsymbol{\alpha}} -\frac{1}{4}\|\boldsymbol{\alpha}\|_2^2 - \frac{1}{4\lambda}\|X^\top \boldsymbol{\alpha}\|_2^2 + \boldsymbol{\alpha}^\top \mathbf{y}. \tag{20}$$

Since there is no constraint on $\boldsymbol{\alpha}$, we can again take the derivative w.r.t. $\boldsymbol{\alpha}$ and set it to zero:

$$-\frac{1}{2}\boldsymbol{\alpha} - \frac{1}{2\lambda} XX^\top \boldsymbol{\alpha} + \mathbf{y} = 0. \tag{21}$$

Thus, we have

$$\boldsymbol{\alpha} = 2\lambda (XX^\top + \lambda \mathbb{I}_n)^{-1} \mathbf{y}, \tag{22}$$

and hence follows from (18) that

$$\mathbf{w}^\star = X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1}\mathbf{y}. \tag{23}$$

2. (15 pts) Based on the formula (16), derive an algorithm for kernel ridge regression, i.e., replace each row $\mathbf{x}_i$ by $\phi(\mathbf{x}_i)$, where ϕ is the feature transform of some kernel function $\kappa(\mathbf{x}, \mathbf{x}')$. Please specify what your algorithm will maintain and how to compute the response $\hat{y}$ on a new test sample $\mathbf{x}$. Analyze the space and run time complexity (both training and test) of your algorithm. [Under no circumstance should your algorithm depend **explicitly** on the feature transform ϕ.]

   Ans: Let $K \in \mathbb{R}^{n\times n}$ with $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Then, it follows from (16) that in the feature space:

$$\mathbf{w}^\star = \phi(X)^\top (K + \lambda \mathbb{I}_n)^{-1}\mathbf{y}, \tag{24}$$

   where $\phi(X) = [\phi(\mathbf{x}_1)^\top, \ldots, \phi(\mathbf{x}_n)^\top]^\top$. The algorithm for kernelized ridge regression will maintain $\boldsymbol{\alpha} := (K + \lambda \mathbb{I}_n)^{-1}\mathbf{y}$, which is in $\mathbb{R}^n$. For testing, upon receiving a test point $\mathbf{x}^*$, we compute the n-dimensional vector $\mathbf{k}^*$ with $k^*_i := \kappa(\mathbf{x}_i, \mathbf{x}^*)$. Then, the response y^* on $\mathbf{x}^*$ is $(\mathbf{k}^*)^\top (K + \lambda \mathbb{I}_n)^{-1}\mathbf{y}$.

   For training, we need to compute the $n \times n$ matrix K and solve the linear system $(K + \lambda \mathbb{I}_n)^{-1}\mathbf{y}$, in total costing time $O(n^3 + n^2 d)$ (assuming it costs $O(d)$ to evaluate the kernel κ) and space $O(n^2 + d)$. For testing, assuming we already have $\boldsymbol{\alpha} := (K + \lambda \mathbb{I}_n)^{-1}\mathbf{y}$, then it costs time $O(nd)$ and space $O(n + d)$ since we need to compute $\mathbf{k}^*$ and the dot product $\boldsymbol{\alpha}^\top \mathbf{k}^*$.

3. (15 pts) From the equivalence between (13) and (16) prove the Sherman-Morrison formula:

$$(X^\top X + \lambda \mathbb{I}_d)^{-1} = \tfrac{1}{\lambda}\mathbb{I}_d - \tfrac{1}{\lambda} X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1} X. \tag{25}$$

   [This result is very useful in performing rank-1 updates in many ML algorithms: when $n = 1$, the left-hand side costs $O(d^3)$ while the right-hand side only costs $O(d^2)$.]

   Ans: The equivalence gives us:

$$(X^\top X + \lambda \mathbb{I}_d)^{-1} X^\top \mathbf{y} = X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1}\mathbf{y}. \tag{26}$$

   Since this is true for any $\mathbf{y}$, we then have

$$(X^\top X + \lambda \mathbb{I}_d)^{-1} X^\top = X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1}. \tag{27}$$

   Multiplying both ends from the right by X:

$$(X^\top X + \lambda \mathbb{I}_d)^{-1} X^\top X = X^\top (XX^\top + \lambda \mathbb{I}_n)^{-1} X. \tag{28}$$

   Further, for the left-hand side:

$$(X^\top X + \lambda \mathbb{I}_d)^{-1} X^\top X = (X^\top X + \lambda \mathbb{I}_d)^{-1}(X^\top X + \lambda \mathbb{I}_d - \lambda \mathbb{I}_d) = \mathbb{I}_d - \lambda (X^\top X + \lambda \mathbb{I}_d)^{-1}. \tag{29}$$

   Combining (28) and (29) and rearranging we arrive at (25).

Exercise 3: Kernels (15 pts)

Note: This exercise is best done by strictly following the order. If you get stuck on a subproblem, you can assume its validity for the following subproblems.

1. (5 pts) Prove that if for each n, $\kappa_n : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a kernel, then $\kappa := \lim_{n\to\infty} \kappa_n$ (assuming the pointwise limit exists) is again a kernel. [$\lim_n (\alpha a_n + \beta b_n) = \alpha \lim_n a_n + \beta \lim_n b_n$ when all limits exist.]

   Ans: For any m, and any $\mathbf{x}_1, \ldots, \mathbf{x}_m$, we know the $m \times m$ matrix $K^{(n)}$ with $K^{(n)}_{ij} := \kappa_n(\mathbf{x}_i, \mathbf{x}_j)$ is symmetric and positive semidefinite, since κ_n is a kernel. Define the $m \times m$ matrix K with $K_{ij} =$

$\lim_n K_{ij}^{(n)} = \lim_n \kappa_n(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Clearly K is symmetric. It is also positive semidefinite since for any $\boldsymbol{\alpha} \in \mathbb{R}^m$:

$$\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j K_{ij} = \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \lim_n K_{ij}^{(n)} = \lim_n \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j K_{ij}^{(n)} \geq 0, \tag{30}$$

since for any n, $\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j K_{ij}^{(n)} \geq 0$. Therefore, κ is indeed a kernel.

2. (5 pts) Prove that if $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a kernel, then $\exp(\kappa)$ is again a kernel. [Use the Taylor expansion of the exponential function.]

   Ans: By definition, $\exp(\kappa) = \sum_{i=1}^\infty \frac{\kappa^i}{i!}$. As shown in class, $\kappa_n := \sum_{i=1}^n \frac{\kappa^i}{i!}$ is a kernel for any n since κ is. Thus, $\exp(\kappa) = \lim_n \kappa_n$ is also a kernel, thanks to the previous exercise.

3. (5 pts) Prove that the Gaussian density $\kappa(\mathbf{x}, \mathbf{x}') := \exp(-\|\mathbf{x} - \mathbf{x}'\|_2^2/\sigma)$ is a kernel for any $\sigma > 0$. [Try to expand the squared norm $\|\mathbf{x} - \mathbf{x}'\|_2^2$ and break it into three terms.]

   Ans: For any m and any $\mathbf{x}_1, \ldots, \mathbf{x}_m$, let K be the $m \times m$ matrix with $K_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma)$. Clearly, K is symmetric. To see that K is also positive semidefinite, let $\boldsymbol{\alpha} \in \mathbb{R}^m$, then

$$\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j K_{ij} = \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma) \tag{31}$$

$$= \sum_{i=1}^m \sum_{j=1}^m [\alpha_i \exp(-\|\mathbf{x}_i\|_2^2/\sigma)][\alpha_j \exp(-\|\mathbf{x}_j\|_2^2/\sigma)] \exp(2\mathbf{x}_i^\top \mathbf{x}_j/\sigma). \tag{32}$$

   Note that the function $\kappa(\mathbf{x}, \mathbf{x}') := \mathbf{x}^\top \mathbf{x}'$ is trivially a kernel (consider the feature transform $\phi(\mathbf{x}) = \mathbf{x}$). Hence, by the previous exercise we know $\exp(2\kappa/\sigma)$ is also a kernel. Thus, on the dataset $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, the kernel matrix $\tilde{K}$ with $\tilde{K}_{ij} = \exp(2\mathbf{x}_i^\top \mathbf{x}_j/\sigma)$ is symmetric and positive semidefinite. Hence, for any $\boldsymbol{\beta} \in \mathbb{R}^m$, we have

$$\sum_{i=1}^m \sum_{j=1}^m \beta_i \beta_j \exp(2\mathbf{x}_i^\top \mathbf{x}_j/\sigma) \geq 0. \tag{33}$$

   Our proof is now complete by setting β_i in the above to $\alpha_i \exp(-\|\mathbf{x}_i\|_2^2/\sigma)$ and concluding that (32) is indeed nonnegative, i.e., K is positive semidefinite.