Homework #6

RELEASE DATE: 11/18/2024

DUE DATE: 12/02/2024, BEFORE 13:00 on GRADESCOPE

QUESTIONS ARE WELCOMED ON DISCORD (INFORMALLY) OR VIA EMAILS (FORMALLY).

You will use Gradescope to upload your scanned/printed solutions. Any programming language/platform is allowed.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

This homework set comes with 200 points and 20 bonus points. In general, every homework set would come with a full credit of 200 points, with some possible bonus points.

Experimentally, we will allow each grading TA to give up to 2 clarity-bonus points for every human-graded problem if the TA believes that the answer is delivered with exceptional clarity, in addition to being correct. We hope that this encourages everyone to think about how to express your ideas clearly.

1. (10 points, auto-graded) In this problem, we are going to apply the kernel trick to the perceptron learning algorithm introduced in Machine Learning Foundations. If we run the perceptron learning algorithm on the transformed examples $\{(\mathbf{\Phi}(\mathbf{x}_n), y_n)\}_{n=1}^N$, the algorithm updates $\mathbf{w}_t$ to $\mathbf{w}_{t+1}$ when the current $\mathbf{w}_t$ makes a mistake on $\Big(\mathbf{\Phi}(\mathbf{x}_{n(t)}), y_{n(t)}\Big)$:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_{n(t)}\mathbf{\Phi}(\mathbf{x}_{n(t)})$$

Because every update is based on one (transformed) example, if we take $\mathbf{w}_0 = \mathbf{0}$, we can represent every $\mathbf{w}_t$ as a linear combination of $\{\mathbf{\Phi}(\mathbf{x}_n)\}_{n=1}^N$. We can then maintain the linear combination coefficients instead of the whole $\mathbf{w}$. Assume that we maintain an N-dimensional vector $\boldsymbol{\alpha}_t$ in the t-th iteration such that

$$\mathbf{w}_t = \sum_{n=1}^{N} \alpha_{t,n}\mathbf{\Phi}(\mathbf{x}_n)$$

for $t = 0, 1, 2, \ldots$. Set $\boldsymbol{\alpha}_0 = \mathbf{0}$ (N zeros) to match $\mathbf{w}_0 = \mathbf{0}$ ($\tilde{d}+1$ zeros). How should $\boldsymbol{\alpha}_t$ be updated to $\boldsymbol{\alpha}_{t+1}$ when the current $\mathbf{w}_t$ (represented by $\boldsymbol{\alpha}_t$) makes a mistake on $\Big(\mathbf{\Phi}(\mathbf{x}_{n(t)}), y_{n(t)}\Big)$? Choose the correct answer.

[a] $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t$ except $\alpha_{t+1,n(t)} \leftarrow \alpha_{t,n(t)} + y_{n(t)}$

[b] $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t$ except $\alpha_{t+1,n(t)} \leftarrow \alpha_{t,n(t)} - y_{n(t)}$

[c] $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t$ except $\alpha_{t+1,n(t)} \leftarrow \alpha_{t,n(t)} + 1$

[d] $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t$ except $\alpha_{t+1,n(t)} \leftarrow \alpha_{t,n(t)} - 1$

[e] $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t + \mathbf{y}$

2. (10 points, auto-graded) Consider the soft-margin SVM taught in our class. Assume that after solving the dual problem, every example is a bounded support vector. That is, the optimal solution $\boldsymbol{\alpha}^*$ satisfies $\boldsymbol{\alpha}_n^* = C$ for every example. In this case, there may be multiple solutions for the optimal b^* for the primal SVM problem. What is the smallest such b^*? Choose the correct answer.

[a] $\min\limits_{n=1,2,\ldots,N} \left(y_n - y_n\left(\sum_{m=1}^N y_m \alpha_m^* K(x_n, x_m)\right)\right)$

[b] $\min\limits_{n:\ y_n>0} \left(1 - \left(\sum_{m=1}^N y_m \alpha_m^* K(x_n, x_m)\right)\right)$

[c] $\max\limits_{n:\ y_n>0} \left(1 - \left(\sum_{m=1}^N y_m \alpha_m^* K(x_n, x_m)\right)\right)$

[d] $\min\limits_{n:\ y_n<0} \left(-1 - \left(\sum_{m=1}^N y_m \alpha_m^* K(x_n, x_m)\right)\right)$

[e] $\max\limits_{n:\ y_n<0} \left(-1 - \left(\sum_{m=1}^N y_m \alpha_m^* K(x_n, x_m)\right)\right)$

3. (10 points, auto-graded)In class, we learned about how to extend the non-linear soft-margin SVM to a squared hinge loss one as follows, which penalizes the margin violation quadratically.

$$\begin{aligned} (P_2) \quad \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^N \xi_n^2 \\ \text{subject to} \quad & y_n\Big(\mathbf{w}^T\boldsymbol{\Phi}(\mathbf{x}_n) + b\Big) \geq 1 - \xi_n, \text{ for } n = 1, 2, \ldots, N. \end{aligned}$$

The dual problem of (P_2) will look like this:

$$\begin{aligned} (D_2) \quad \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{n=1}^N\sum_{m=1}^N \alpha_n\alpha_m y_n y_m \cdot \left(K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{2C}[\![n = m]\!]\right) - \sum_{n=1}^N \alpha_n \\ \text{subject to} \quad & \sum_{n=1}^N y_n\alpha_n = 0 \\ & \alpha_n \geq 0, \text{ for } n = 1, 2, \ldots, N, \end{aligned}$$

where the kernel function $K(\mathbf{x}, \mathbf{x}') = \boldsymbol{\Phi}(\mathbf{x})^T\boldsymbol{\Phi}(\mathbf{x}')$. After getting the optimal $\boldsymbol{\alpha}^*$ for (D_2), how can we calculate the optimal $\boldsymbol{\xi}^*$ for (P_2)? Choose the correct answer.

[a] $\boldsymbol{\xi}^* = \boldsymbol{\alpha}^*$

[b] $\boldsymbol{\xi}^* = C \cdot \mathbf{1} - \boldsymbol{\alpha}^*$

[c] $\boldsymbol{\xi}^* = C \cdot \mathbf{1} + \boldsymbol{\alpha}^*$

[d] $\boldsymbol{\xi}^* = \frac{1}{2}\boldsymbol{\alpha}^*$

[e] $\boldsymbol{\xi}^* = \frac{1}{2C}\boldsymbol{\alpha}^*$

4. (10 points, auto-graded) For a binary classification task, assume that there are 5 binary classifiers g_1, g_2, …, g_5, and for some $P(\mathbf{x}, y)$, the errors made by the 5 classifiers are independent. That is, the five random variables $[\![y \neq g_1(\mathbf{x})]\!]$, $[\![y \neq g_2(\mathbf{x})]\!]$, …, $[\![y \neq g_5(\mathbf{x})]\!]$ are independent. Assume that $E_{\text{out}}(g_t) = 0.25$ for $t = 1, 2, \ldots, 5$, if uniform blending is used to blend the five classifiers to get G like Page 7 of Lecture 207, what is $E_{\text{out}}(G)$? Choose the closest answer.

[a] 0.25

[b] 0.20

[c] 0.15

[d] 0.10

[e] 0.05

5. (20 points, human-graded) Consider the linear soft-margin SVM discussed in class.

$$\begin{aligned}(P_1) \quad \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n \\ \text{subject to} \quad & y_n\Big(\mathbf{w}^T\mathbf{x}_n + b\Big) \geq 1 - \xi_n, \text{ for } n = 1, 2, \ldots, N. \\ & \xi_n \geq 0, \text{ for } n = 1, 2, \ldots, N.\end{aligned}$$

The dual problem of (P_1) will look like this:

$$\begin{aligned}(D_1) \quad \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \mathbf{x}_n^T\mathbf{x}_m - \sum_{n=1}^{N}\alpha_n \\ \text{subject to} \quad & \sum_{n=1}^{N} y_n\alpha_n = 0 \\ & \alpha_n \geq 0, \text{ for } n = 1, 2, \ldots, N,\end{aligned}$$

Recall that we separated the shorter $\mathbf{w}$ from b to derive (P_1). Nevertheless, Dr. Threshold totally forgot about this when solving the SVM. That is, Dr. Threshold fed $\mathbf{z}_n^T = [1, \mathbf{x}_n^T]$ when solving (P_1) or (D_1) instead of $\mathbf{x}_n^T$. Assume that the primal-dual solution that Dr. Threshold obtains from the problem can be represented as $(b^*, \tilde{\mathbf{w}}^*, \boldsymbol{\alpha}^*)$, where $\tilde{\mathbf{w}}^* \in \mathbb{R}^{d+1}$ is a longer vector with the same dimension as any $\mathbf{z}_n$. For simplicity, you can assume that the solution is unique. Let $\mathbf{w}^* = [\tilde{w}_1^*, \tilde{w}_2^*, \ldots, \tilde{w}_d^*]^T$. Prove or disprove that $(b^*, \mathbf{w}^*, \boldsymbol{\alpha}^*)$ is also an optimal solution of the original problem (which contains shorter $\mathbf{x}_n$) and $\tilde{w}_0^* = 0$.

6. (20 points, human-graded) The soft-margin SVM that we introduced in class are for binary classification. Researchers have attempted to extend it to a one-class formulation for outlier detection, using only examples of $y_n = +1$. In particular, when given $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, the goal of the one-class formulation is to find a hyperplane such that most of the examples are "normally" on one side of the hyperplane and some of the outliers are on the other side.

There is a trivial solution to the task above—simply put the hyperplane far far away from any examples! So the one-class problem is not as easy as it sounds. Anyway, one simple way to extend the binary-classification SVM to a one-class formulation is to consider an anchor pseudo-example that is strictly negative. In this problem, we will take $\mathbf{0}$, the origin of the space, as such an anchor pseudo-example $\mathbf{x}_0 = \mathbf{0}$ with $y_0 = -1$. Because of the specialty of the anchor example, it needs to satisfy the hard-margin constraint (without any ξ_0), while other examples only need to satisfy the soft-margin constraints. That is, the primal problem is

$$\begin{aligned}(P) \quad \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n \\ \text{subject to} \quad & y_n\Big(\mathbf{w}^T\boldsymbol{\Phi}(\mathbf{x}_n) + b\Big) \geq 1 - \xi_n, \text{ for } n = 1, 2, \ldots, N. \\ & \xi_n \geq 0, \text{ for } n = 1, 2, \ldots, N. \\ & y_0\Big(\mathbf{w}^T\boldsymbol{\Phi}(\mathbf{x}_0) + b\Big) \geq 1.\end{aligned}$$

Note that $y_0 = -1$ and other y_n's are $+1$. Derive the Lagrange dual problem of (P) that involves *only* the N Lagrange multipliers of the constraints on $(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$, …, $(\mathbf{x}_N, y_N)$. The dual problem needs to be expressed in a standard convex quadratic programming form with $\mathrm{Q}, \mathbf{p}, \mathrm{A}, \mathrm{c}$, like the typical dual problem of the soft-margin SVM.

7. (20 points, human-graded) For a set of examples $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ with $N \geq 2$ and a kernel function K, consider a hypothesis set that contains

$$h_{\boldsymbol{\alpha},b}(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} y_n \alpha_n K(\mathbf{x}_n, \mathbf{x}) + b\right).$$

The classifier returned by SVM can be viewed as one such $h_{\boldsymbol{\alpha},b}$, where the values of $\boldsymbol{\alpha}$ is determined by the dual QP solver and b is calculated from the KKT conditions.

In this problem, we will consider the Gaussian kernel $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$. We will study a simpler form of $h_{\boldsymbol{\alpha},b}$ where $\boldsymbol{\alpha} = \mathbf{1}$ (the vector of all 1's) and $b = 0$. Let us name $h_{\mathbf{1},0}$ as $\hat{h}$ for simplicity. Assume that the distance between any pair of different $(\mathbf{x}_n, \mathbf{x}_m)$ in the $\mathcal{X}$-space is no less than ϵ. Prove that when $\gamma > \frac{\ln(N-1)}{\epsilon^2}$, $E_{\text{in}}(\hat{h}) = 0$. That is, when using the Gaussian kernel, we can "easily" separate the given data set if γ is large enough.

8. (20 points, human-graded) Prove that

$$K(x, x') = \exp\big(2\cos(x - x') - 2\big)$$

is a valid kernel for $x \in \mathbb{R}$ and $x' \in \mathbb{R}$. Note that the kernel has an interesting property of measuring the similarity between x and x' periodically. You can use the results from high-school math, any typical linear algebra textbook, and any facts that has been introduced in class, e.g. $\cos(x - x') = \cos x \cos x' + \sin x \sin x'$, or the multi-dimensional Gaussian kernel derived in class is a valid kernel.

9. (20 points, human-graded) When talking about non-uniform voting in aggregation, we mentioned that $\boldsymbol{\alpha}$ can be viewed as a weight vector learned from any linear algorithm coupled with the following transform:

$$\boldsymbol{\Phi}(\mathbf{x}) = \Big(g_1(\mathbf{x}), g_2(\mathbf{x}), \cdots, g_T(\mathbf{x})\Big).$$

When studying kernel methods, we mentioned that the kernel is simply a computational short-cut for the inner product $(\boldsymbol{\Phi}(\mathbf{x}))^T(\boldsymbol{\Phi}(\mathbf{x}'))$. In this problem, we mix the two topics together using the decision stumps as our $g_t(\mathbf{x})$.

Assume that the input vectors contain only integers between (including) L and R, where $L < R$. Consider some *scaled* decision stumps $g_{i,\theta}(\mathbf{x}) = [\![x_i > \theta]\!]$, where

$$\begin{array}{ll} i & \in \{1, 2, \cdots, d\}, \\ d & \text{is the finite dimensionality of the input space,} \\ \theta & \in \{\theta_1 = L + 0.5, \theta_2 = L + 1.5, \ldots, \theta_k = R - 0.5\}. \end{array}$$

The scaled decision stumps look like a sharper form of the sigmoid function that outputs in $[0, 1]$ (actually, $\{0, 1\}$). Define $\boldsymbol{\Phi}_{ds}(\mathbf{x}) = \Big[g_{1,\theta_1}(\mathbf{x}), g_{1,\theta_2}(\mathbf{x}), \ldots, g_{1,\theta_k}(\mathbf{x}), \ldots, g_{d,\theta_k}(\mathbf{x})\Big]^T$ as a column vector. What is $K_{ds}(\mathbf{x}, \mathbf{x}') = (\boldsymbol{\Phi}_{ds}(\mathbf{x}))^T(\boldsymbol{\Phi}_{ds}(\mathbf{x}'))$? Prove your answer. The TAs will take the simplicity of your kernel into account during grading.

(Hint: This result shows that aggregation learning with SVMs is possible. Those who are interested in knowing how perceptrons and decision trees can be used instead of decision stumps during kernel construction can read the following early work of you-know-who.)

`https://www.csie.ntu.edu.tw/~htlin/paper/doc/infkernel.pdf`

10. (20 points, human-graded) For Problems 10 to 12, we are going to experiment with a real-world data set. We will reuse the `mnist.scale` dataset at

`https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass/mnist.scale.bz2`

for training. The dataset is originally for multi-class classification. We will study one of the sub-problems within one-versus-one decomposition: class 3 versus class 7.

Some quadratic programming packages cannot handle such a large problem. We recommend that you consider the LIBSVM package

`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Regardless of the package that you choose to use, please read the manual of the package carefully to make sure that you are indeed solving the soft-margin support vector machine taught in class like the dual formulation below:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m K(\mathbf{x}_n,\mathbf{x}_m) - \sum_{n=1}^{N}\alpha_n \\ \text{subjectto} \quad & \sum_{n=1}^{N} y_n\alpha_n = 0 \\ & 0 \le \alpha_n \le C \quad n = 1,\ldots,N. \end{aligned}$$

Some practical remarks include

(i) Please tell your chosen package to **not** automatically scale the data for you, lest you should change the effective kernel and get different results.

(ii) It is your responsibility to check whether your chosen package solves the designated formulation with enough numerical precision. Please read the manual of your chosen package for software parameters whose values affect the outcome—any ML practitioner needs to deal with this kind of added uncertainty.

Consider the polynomial kernel $K(\mathbf{x}_n,\mathbf{x}_m) = (1+\mathbf{x}_n^T\mathbf{x}_m)^Q$, where Q is the degree of the polynomial. Among $C \in \{0.1, 1, 10\}$ and $Q \in \{2, 3, 4\}$, count the number of support vectors of the resulting SVM classifier. Provide those numbers with a clear table. Which of the (C, Q) combination(s) result in the smallest number of support vectors? Describe your findings. Please include screenshots of the first page of your command or code.

11. (20 points, human-graded) Consider the Gaussian kernel $K(\mathbf{x}_n,\mathbf{x}_m) = \exp\left(-\gamma||\mathbf{x}_n-\mathbf{x}_m||^2\right)$. Among $C \in \{0.1, 1, 10\}$ and $\gamma \in \{0.1, 1, 10\}$, calculate $1/\|\mathbf{w}\|$, which is the margin, at the optimal solution with the kernel trick. Provide those numbers with a clear table. Which of the (C, γ) combination(s) result in the largest margin? Describe your findings. Please include screenshots of the first page of your command or code.

12. (20 points, human-graded) Following Problem 11 and consider a validation procedure that randomly samples 200 examples from the training set for validation and leaves the other examples for training g_{SVM}^{-}. Fix $C = 1$ and use the validation procedure to choose the best γ among $\{0.01, 0.1, 1, 10, 100\}$ according to E_{val}, evaluated with the 0/1 error. If there is a tie of E_{val}, choose the smallest γ. Repeat the procedure 128 times and count the number of times that each γ is selected. Plot a bar chart of γ versus its selection frequency. Describe your findings. Please include screenshots of the first page of your command or code.

13. (Bonus 20 points, human-graded) Derive a simplified Lagrange dual problem of the soft-margin SVM dual. Is your Lagrange dual problem of the soft-margin SVM dual the same as the soft-margin SVM primal? Are they "similar" in any sense? Describe your findings.

For your information, this is what chatGPT says.