EE 559 **Homework 6 (Week 10)** Posted: Sat., 3/26/2022
Jenkins Due: Sat. 4/2/2022, 1:00 PM PDT

v2 adds a tip on Pr. 2(g)

Please note:

1. *Reminder:* Python only for coding.
2. *Coding quality.* Starting with this homework, your coding quality will be graded and will count towards your homework score. You will probably find that this homework involves somewhat less coding than the typical homework assignment. You are still encouraged to apply good coding techniques; it is a good opportunity to put it into practice.

   Jupyter notebooks are allowed; however it can take some extra effort at first to use good coding practices in the notebook format (until you get used to it). You are expected to write code of good quality whether you use the notebook format or .py files.

   Guidelines on coding quality can be found in Discussion 6 (Week 7) and related documents posted on D2L in Week 7.
3. *Files to upload.* As usual, please submit 2 pdf files: one of all your answers, and one computer-readable pdf of all your code.

1. This problem is to be done by hand. In the parts below, you will set up the primal Lagrangian and derive the dual Lagrangian, for support vector machine classifier, for the case of data that is linearly separable in expanded feature space.

   For the SVM learning, we stated the optimization problem as:

$$\text{Minimize } J\left(\underline{w}\right) = \tfrac{1}{2}\left\|\underline{w}\right\|^2$$
$$\text{s.t. } z_i\left(\underline{w}^T \underline{u}_i + w_0\right) - 1 \geq 0 \quad \forall i$$

   Assume our data is linearly separable in the expanded feature space. Also, nonaugmented notation is used throughout this problem.

(a) If the above set of constraints (second line of equations above) is satisfied, will all the training data be correctly classified?

(b) Write the Lagrangian function $L\left(\underline{w}, w_0, \underline{\lambda}\right)$ for the minimization problem stated above. Use $\lambda_i, i = 1, 2, \cdots, N$ for the Lagrange multipliers. Also state the KKT conditions. **(Hint:** there are 3 KKT conditions).

(c) Derive the dual Lagrangian L_D, by proceeding as follows:

   (i) Minimize L w.r.t. the weights.

Hint: solve $\nabla_{\underline{w}} L = \underline{0}$ for the optimal weight $\underline{w}^*$ (in terms of λ_i and other variables); and set $\frac{\partial L}{\partial w_0} = 0$ and simplify.

(ii) Substitute your expressions from part (i) into L, and use your expression from $\frac{\partial L}{\partial w_0} = 0$ as a new constraint, to derive L_D as:

$$L_D(\underline{\lambda}) = -\frac{1}{2}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j z_i z_j \underline{u}_i{}^T \underline{u}_j\right] + \sum_{i=1}^{N}\lambda_i$$

subject to the (new) constraint: $\sum_{i=1}^{N}\lambda_i z_i = 0$, which becomes a new KKT condition. Also give the other two KKT conditions on λ_i, which carry over from the primal form.

2. In this problem you will do SVM learning on a small set of given data points, using the result from Problem 1 above. Problem 2 also uses nonaugmented notation.

Coding. This problem involves some work by hand, and some solution by coding. As in previous homework assignments, in this problem you are required to write the code yourself; you may use Python built-in functions, NumPy, and matplotlib; and you may use pandas only for reading and/or writing csv files. For the plotting, we are not supplying plotting function(s) for you; may use this as an opportunity to (learn, explore, and) use matplotlib functions as needed. Alternatively, you may do the plots by hand if you prefer.

Throughout this problem, let the dataset have $N = 3$ points. We will work entirely in the expanded feature space ($\underline{u}$-space).

(a) Derive by hand a set of equations, with $\underline{\lambda}$, μ as variables, and $z_i, \underline{u}_i$, $i = 1,2,3$ as given constants, that when solved will give the SVM decision boundary and regions. To do this, start the Lagrangian process to optimize L_D w.r.t. $\underline{\lambda}$ and μ, subject to the equality constraint $\sum_{i=1}^{N}\lambda_i z_i = 0$. Set all the derivatives equal to 0 to obtain the set of equations.

Hint: Start from $L_D{}'(\underline{\lambda},\mu) = \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j z_i z_j \underline{u}_i{}^T \underline{u}_j\right] + \mu\left(\sum_{i=1}^{N} z_i\lambda_i\right)$, in which the last term has been added to incorporate the equality constraint stated above.

Finally, formulate your set of equations as a matrix-vector equation of the form:

$$\underline{\underline{A}}\,\underline{\rho} = \underline{b}$$

in which $\underline{\rho} = \left[\underline{\lambda}^{\mathrm{T}},\ \mu\right]^{\mathrm{T}}$. Give your expressions for $\underline{\underline{A}}$ and $\underline{b}$.

Tip: If you'd like a quick check to see if you're on the right track, try plugging in (by hand) for this simple dataset:

$$u_1 = \begin{bmatrix}1\\0\end{bmatrix},\ u_2 = \begin{bmatrix}0\\0\end{bmatrix} \in S_1\ \ (z_1 = z_2 = 1); \qquad u_3 = \begin{bmatrix}1\\1\end{bmatrix} \in S_2\ (z_3 = -1).$$

The first row of $\underline{\underline{A}}$ should be $[1\ \ 0\ \ -1\ \ -1]$ and first entry of $\underline{b}$ should be 1.

In the parts below you will use a computer to solve this matrix equation for $\underline{\lambda}$ and μ, calculate then plot and interpret the results, for 3 different datasets.

For parts (b)-(e) below, use the following dataset:

$$u_1 = \begin{bmatrix}1\\2\end{bmatrix}, u_2 = \begin{bmatrix}2\\1\end{bmatrix} \in S_1$$

$$u_3 = \begin{bmatrix}0\\0\end{bmatrix} \in S_2$$

For all parts below, consider S_1 to be the positive class ($z_i = +1$).

(b) Write code in Python to solve for $\underline{\lambda}$ and μ, calculate $\underline{w}^*$ and w_0 , and check the KKT conditions.

Tip: Use variables z_i, $\underline{u}_i$, i = 1,2,3 such that you can input their values for different datasets.

Specifically, the code should:

(i) Use NumPy to invert your matrix, and to calculate the resulting values for $\underline{\lambda}$ and μ.

(ii) Check that the resulting $\underline{\lambda}$ satisfies the KKT conditions involving $\underline{\lambda}$ (but not involving $\underline{w}$) that you stated in Problem 1(c)(ii).

(iii) Calculate the optimal (nonaugmented) weight vector $\underline{w}^*$ by using your result from Problem 1(c)(i). And, find the optimal bias term w_0 using one of the KKT conditions from Problem 1(b).

(iv) Check that the resulting $\underline{w}$ and w_0 satisfy the KKT conditions on $\underline{w}$ and w_0 of Pr. 1(c).

(v) Output the results of (i)-(iv) above.

(c) Run your code on the given dataset; give the resulting values for $\underline{\lambda}$ and μ, and the results of the KKT-conditions check on $\underline{\lambda}$.

Also give the resulting $\underline{w}^*$ and w_0, and the results of the KKT-conditions check on $\underline{w}$ and w_0.

(d) Plot in 2D nonaugmented feature ($\underline{u}$) space: the data points showing their class labels, the decision boundary defined by $\underline{w}^*$ and w_0, and an arrow showing which side of the boundary

is class S_1. While you are encouraged to do (some or all of) the plot by computer, you can do some or all of it by hand if you prefer, as long as it is neat and reasonably accurate.

(e) Looking at the plot, does the decision boundary correctly classify the training data?

And if yes, does it look like the *maximum-margin* boundary that will correctly classify the data? Briefly justify.

(f)(i)-(iii) Repeat parts (c) – (e) except for the following dataset:

$$u_1 = \begin{bmatrix}1\\2\end{bmatrix}, u_2 = \begin{bmatrix}2\\1\end{bmatrix} \in S_1$$

$$u_3{}' = \begin{bmatrix}1\\1\end{bmatrix} \in S_2$$

in which the third data point has been changed.

(iv) Explain the change, or lack of change, in the decision boundary from (d) to (f).

(g) (i) How do you think the boundary will change (relative to (f)) if we instead use the following data:

$$u_1 = \begin{bmatrix}1\\2\end{bmatrix}, u_2 = \begin{bmatrix}2\\1\end{bmatrix} \in S_1$$

$$u_3{}'' = \begin{bmatrix}0\\1.5\end{bmatrix} \in S_2$$

in which the third data point has (yet again) been changed?

(ii)-(iv) Try it by repeating (c)-(e) except with these data points[1].

(v) Explain any change or lack of change in the decision boundary as compared with (d) and (f).

[1] **Tip:** Note that the linear algebra approach we take to solve this problem, has no way of enforcing the requirement $\lambda_i \geq 0 \ \forall i$. After you check this requirement, if any data point $\underline{u}_k$ has $\lambda_k < 0$, then you can set $\lambda_k = 0$ as a given constant, and then re-optimize the other Lagrange multipliers. Then check the KKT conditions again to verify they are satisfied.

Why? With no nonnegativity condition on λ, the optimal solution effectively finds a point that is on the boundary of all inequality constraint regions (as in case (b) of Lecture 12 p. 10). If one of the constraints (say on data point $\underline{u}_k$) is already satisfied (as in case (a) on p. 9), it proceeds as in case (b) to find a point on the boundary of the constraint region, which results in a negative Lagrange multiplier $\lambda_k < 0$. Resetting it to 0, then re-optimizing the other parameters, is a way of enforcing the $\lambda_k \geq 0$ requirement.

Comment: this method of implementation (Pr. 2 using matrix inverse) is useful for working with algebraic expressions, theory, and seeing/verifying how things work. For larger datasets, typically other implementation methods are used such as quadratic programming or variations on gradient descent designed for this type of optimization.