Your Name
email@fas.harvard.edu
CS181-S17

Assignment #3
Due: 5:00pm Mar 24th, 2017

Collaborators: John Doe, Fred Doe

SOLUTION - Do Not Distribute
Homework 3: Max-Margin and SVM

Grading Instructions

In the solutions, you will see several highlighted checkpoints. These each have a label that corresponds to an entry in the Canvas quiz for this problem set. The highlighted statement should clearly indicate the criteria for being correct on that problem. If you satisfy the criteria for a problem being correct, mark "Yes" on the corresponding position on the Canvas quiz. Otherwise, mark "No". Your homework scores will be verified by course staff at a later date.

Introduction

This homework assignment will have you work with max-margin methods and SVM classification. The aim of the assignment is (1) to further develop your geometrical intuition behind margin-based classification and decision boundaries, (2) to explore the properties of kernels and how they provide a different form of feature development from basis functions, and finally (3) to implement a basic Kernel based classifier.

There is a mathematical component and a programming component to this homework. Please submit your PDF and Python files to Canvas, and push all of your work to your Github repository. If a question requires you to make any plots, like Problem 3, please include those in the writeup.

Problem 1 (Fitting an SVM by hand, 7pts)

For this problem you will solve an SVM without the help of a computer, relying instead on principled rules and properties of these classifiers.

Consider a dataset with the following 7 data points each with $x \in \mathbb{R}$:

$$\{(x_i, y_i)\}_i = \{(-3, +1), (-2, +1), (-1, -1), (0, -1), (1, -1), (2, +1), (3, +1)\}$$

Consider mapping these points to 2 dimensions using the feature vector $\boldsymbol{\phi}(x) = (x, x^2)$. The hard margin classifier training problem is:

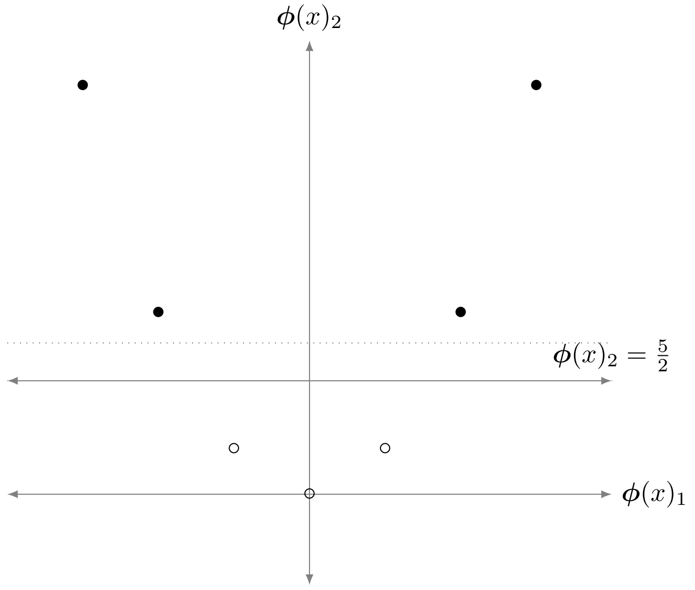
$$\begin{aligned} &\min_{\mathbf{w}, w_0} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \boldsymbol{\phi}(x_i) + w_0) \geq 1, \ \forall i \in \{1, \ldots, n\} \end{aligned} \tag{1}$$

The exercise has been broken down into a series of questions, each providing a part of the solution. Make sure to follow the logical structure of the exercise when composing your answer and to justify each step.

1. Plot the training data in $\mathbb{R}^2$ and draw the decision boundary of the max margin classifer.
2. What is the value of the margin achieved by the optimal decision boundary?
3. What is a vector that is orthogonal to the decision boundary?
4. Considering discriminant $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0) = \mathbf{w}^\top \boldsymbol{\phi}(x) + w_0$, give an expression for *all possible* $(\mathbf{w}, w_0)$ that define the optimal decision boundary. Justify your answer.
5. Consider now the training problem (1). Using your answers so far, what particular solution to $\mathbf{w}$ will be optimal for this optimization problem?
6. Now solve for the corresponding value of w_0, using your general expression from part (4.) for the optimal decision boundary. Write down the discriminant function $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0)$.
7. What are the support vectors of the classifier? Confirm that the solution in part (6.) makes the constraints in (1) binding for support vectors.

Solution

1.



Check 1.1 Your plot matches the one above with $\phi(x)_1, \phi(x)_2$ as axes for the seven points and the correct decision boundary drawn.

2. The margin is the minimum, unsigned, normalized orthogonal distance from any correctly classified point to the decision boundary. In this case, we can consider the unit orthogonal vector $(0, 1)$, and the points $(-2, 4), (-1, 1), (1, 1)$ and $(2, 4)$ (in $\mathbb{R}^2$) are all distance $3/2$ away from the decision boundary (which is defined by $\phi(x)_2 = 5/2$).

   Check 1.2 You gave $\frac{3}{2}$ as the value of the margin.

3. A vector $\mathbf{w}$ that is perpendicular to the decision boundary is $[0\ 1]^\top$.

   Check 1.3 You suggested any vector of the form $[0\ c]^\top$ where $c > 0$.

4. Using vector $\mathbf{w} = [0\ 1]^\top$, and setting discriminant $h(\phi(x); \mathbf{w}, w_0) = \mathbf{w}^\top\phi(x) + w_0 = \phi(x)_2 + w_0 = 0$ for $\phi(x)_2 = 5/2$, we need $w_0 = -5/2$. Now, recognizing that the decision boundary is invariant under scalar multiplication, the set of all possible $(\mathbf{w}, w_0)$ that define the optimal decision boundary is $([0\ \beta]^\top, -\frac{5}{2}\beta)$, with $\beta > 0$.

   Check 1.4 You gave an expression for the possible values of $(\mathbf{w}, w_0)$ equivalent to $([0\ \beta]^\top, -\frac{5}{2}\beta)$, with $\beta > 0$ [for example, $([0\ \frac{2}{3}\beta]^\top, -\frac{5}{3}\beta)$ is equivalent].

5. We want to find the solution that satisfies $y_i(\mathbf{w}^\top\phi(x_i) + w_0) \geq 1$ and minimizes $||\mathbf{w}||_2^2$. For this, we set $y_i(\mathbf{w}^\top\phi(x_i) + w_0) = 1$ on any support vector (these are the examples closest to the decision boundary.) Consider for example $(-2, +1)$ with $\phi(x_2) = [-2\ 4]^\top$, and $y_2 = +1$. We need

$$y_2(\mathbf{w}^\top\phi(x_2) + w_0) = (1)([0\ \beta][-2\ 4]^\top - \frac{5}{2}\beta) = 1,$$

   and thus $4\beta - \frac{5}{2}\beta = 1$, and thus $\beta = \frac{2}{3}$. Therefore, the optimal solution to the training problem has $\mathbf{w} = [0\ \frac{2}{3}]^\top$.

   Check 1.5 Either: you noted that the inequalities in the training problem will be binding on the support vectors in the optimal solution. Based on this you solved for the optimal $\mathbf{w}$ and gave solution $\mathbf{w} = [0\ \frac{2}{3}]^\top$.

(Equivalently): you noted that the margin will be $\frac{1}{\|\mathbf{w}\|}$ in the optimal solution, which based on margin $\frac{3}{2}$ and knowing that $\mathbf{w}$ takes form $[0\ \beta]^\top$ for $\beta > 0$ yields optimal $\mathbf{w} = [0\ \frac{2}{3}]^\top$.

6. We know that $\beta = \frac{2}{3}$, and thus from the general solution $([0\ \beta]^\top, -\frac{5}{2}\beta)$ we have $w_0 = -\frac{5}{2}\beta = -\frac{5}{3}$. The discriminant function is $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0) = [0\ \frac{2}{3}]^\top \boldsymbol{\phi}(x) - 5/3 = \frac{2}{3}\boldsymbol{\phi}(x)_2 - \frac{5}{3} = \frac{2}{3}x^2 - \frac{5}{3}$.

   Check 1.6 You solved for $w_0 = -\frac{5}{3}$ and gave the discriminant $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0) = \frac{2}{3}x^2 - \frac{5}{3}$. [It is also ok to skip the final step of writing $\boldsymbol{\phi}(x)_2$ as x^2.]

7. The support vectors for this problem are the points on the margin boundary (since this is a hard margin classifier), which in the original space are examples $(-2, +1), (-1, -1), (-1, -1), (2, +1)$. Plugging in each of these examples to the discriminant function, we find that $y_i(\mathbf{w}^\top \boldsymbol{\phi}(x_i) + w_0) = 1$ for each of these examples, and thus the constraints are binding.

   Check 1.7 You specified that the support vectors are the examples on the margin boundary: $(-2, +1), (-1, -1), (-1, -1), (2, +1)$, and you showed that $y_i(\mathbf{w}^\top \boldsymbol{\phi}(x_i) + w_0) = 1$ at each of these examples. [it is also ok to specify the support vectors in $\mathbb{R}^2$ space.]

End Solution

Problem 2 (Composing Kernel Functions, 10pts)

A key benefit of SVM training is the ability to use kernel functions $K(\mathbf{x}, \mathbf{x}')$ as opposed to explicit basis functions $\boldsymbol{\phi}(\mathbf{x})$. Kernels make it possible to implicitly express large or even infinite dimensional basis features. We do this by computing $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ directly, without ever computing $\boldsymbol{\phi}(\mathbf{x})$.

When training SVMs, we begin by computing the kernel matrix $\mathbf{K}$, over our training data $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. The kernel matrix, defined as $K_{i,i'} = K(\mathbf{x}_i, \mathbf{x}_{i'})$, expresses the kernel function applied between all pairs of training points.

In class, we saw Mercer's theorem, which tells us that any function K that yields a positive semi-definite kernel matrix forms a valid kernel, i.e. corresponds to a matrix of dot-products under *some* basis $\boldsymbol{\phi}$. Therefore instead of using an explicit basis, we can build kernel functions directly that fulfill this property.

A particularly nice benefit of this theorem is that it allows us to build more expressive kernels by composition. In this problem, you are tasked with using Mercer's theorem and the definition of a kernel matrix to prove that the following compositions are valid kernels, assuming $K^{(1)}$ and $K^{(2)}$ are valid kernels. Recall that a positive semi-definite matrix $\mathbf{K}$ requires $\mathbf{z}^\top \mathbf{K} \mathbf{z} \geq 0, \ \forall \ \mathbf{z} \in \mathbb{R}^n$.

1. $K(\mathbf{x}, \mathbf{x}') = c\, K^{(1)}(\mathbf{x}, \mathbf{x}') \quad \text{for } c > 0$
2. $K(\mathbf{x}, \mathbf{x}') = K^{(1)}(\mathbf{x}, \mathbf{x}') + K^{(2)}(\mathbf{x}, \mathbf{x}')$
3. $K(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})\, K^{(1)}(\mathbf{x}, \mathbf{x}')\, f(\mathbf{x}') \quad$ where f is any function from $\mathbb{R}^m$ to $\mathbb{R}$
4. $K(\mathbf{x}, \mathbf{x}') = K^{(1)}(\mathbf{x}, \mathbf{x}')\, K^{(2)}(\mathbf{x}, \mathbf{x}')$

   [Hint: Use the property that for any $\boldsymbol{\phi}(\mathbf{x})$, $K(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ forms a positive semi-definite kernel matrix.]
5. (a) The exp function can be written as,

   $$\exp(x) = \lim_{i \to \infty} \left(1 + x + \cdots + \frac{x^i}{i!} \right).$$

   Use this to show that $\exp(xx')$ (here $x, x' \in \mathbb{R}$)) can be written as $\boldsymbol{\phi}(x)^\top \boldsymbol{\phi}(x')$ for some basis function $\boldsymbol{\phi}(x)$. Derive this basis function, and explain why this would be hard to use as a basis in standard logistic regression.

   (b) Using the previous identities, show that $K(\mathbf{x}, \mathbf{x}') = \exp(K^{(1)}(\mathbf{x}, \mathbf{x}'))$ is a valid kernel.
6. Finally use this analysis and previous identities to prove the validity of the Gaussian kernel:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||_2^2}{2\sigma^2} \right)$$

Solution

For parts 1-5, it suffices to show that the kernel is valid either by finding a particular $\boldsymbol{\phi}(\mathbf{x})$ that produces it, or by showing that the kernel matrix is positive semi-definite. In these solutions, let $\boldsymbol{\phi}^{(i)}(\mathbf{x})$ and $\mathbf{K}^{(i)}$ denote the underlying basis function and kernel matrix for kernel $K^{(i)}$, respectively.

1. We have the kernel matrix $\mathbf{K} = c\mathbf{K}^{(1)}$. We need $\mathbf{v}^\top \mathbf{K}\mathbf{v} = c\mathbf{v}^\top \mathbf{K}^{(1)}\mathbf{v} \geq 0$, which we know to be true because $\mathbf{K}^{(1)}$ is positive semi-definite and $c > 0$.
   Alternatively, take $\boldsymbol{\phi}(\mathbf{x}) = \sqrt{c}\,\boldsymbol{\phi}^{(1)}(\mathbf{x})$.

   Check 2.1 You correctly proved that the kernel is valid.

2. We have the kernel matrix $\mathbf{K} = \mathbf{K}^{(1)} + \mathbf{K}^{(2)}$. We need

$$\mathbf{v}^\top \mathbf{K}\mathbf{v} = \mathbf{v}^\top(\mathbf{K}^{(1)} + \mathbf{K}^{(2)})\mathbf{v} = \mathbf{v}^\top \mathbf{K}^{(1)}\mathbf{v} + \mathbf{v}^\top \mathbf{K}^{(2)}\mathbf{v} \geq 0,$$

   which we know to be true because $\mathbf{K}^{(1)}, \mathbf{K}^{(2)}$ are positive semi-definite.
   Alternatively, take $\boldsymbol{\phi}(\mathbf{x}) = \left[\phi_1^{(1)}(\mathbf{x}), \ldots, \phi_d^{(1)}(\mathbf{x}), \phi_1^{(2)}(\mathbf{x}), \ldots, \phi_d^{(2)}(\mathbf{x})\right]^\top$, the concatenation of $\boldsymbol{\phi}^{(1)}(\mathbf{x}), \boldsymbol{\phi}^{(2)}(\mathbf{x})$

   Check 2.2 You correctly proved that the kernel is valid.

3. Take $\boldsymbol{\phi}(\mathbf{x}) = f(\mathbf{x})\boldsymbol{\phi}^{(1)}(\mathbf{x})$.

   Check 2.3 You correctly proved that the kernel is valid.

4. Take $\boldsymbol{\phi}(\mathbf{x}) = \left[\phi_1^{(1)}(\mathbf{x})\phi_1^{(2)}(\mathbf{x}), \ldots, \phi_1^{(1)}(\mathbf{x})\phi_d^{(2)}(\mathbf{x}), \phi_2^{(1)}(\mathbf{x})\phi_1^{(2)}(\mathbf{x}), \ldots, \phi_d^{(1)}(\mathbf{x})\phi_d^{(2)}(\mathbf{x})\right]^\top$, the flattened vector for outer product $\boldsymbol{\phi}^{(1)}(\mathbf{x}) \otimes \boldsymbol{\phi}^{(2)}(\mathbf{x})$. The order of terms does not matter.

   Check 2.4 You correctly proved that the kernel is valid.

5. (a) Using the Taylor series expansion, we see that $\exp(xx') = \lim_{i\to\infty}\left(1 + xx' + \cdots + \frac{(xx')^i}{i!}\right)$. Take

$$\boldsymbol{\phi}(x) = \left[1, x, \ldots, \frac{x^i}{\sqrt{i!}}, \ldots\right]^\top.$$

   This is infeasible because we have an infinite basis!

   Note: don't forget the square root on the factorial in the denominator.

   (b) We have

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(K^{(1)}(\mathbf{x}, \mathbf{x}')\right) = \exp\left(\boldsymbol{\phi}^{(1)}(\mathbf{x})^\top \boldsymbol{\phi}^{(1)}(\mathbf{x}')\right) = \prod_{i=1}^{d} \exp\left(\phi_i^{(1)}(\mathbf{x})\phi_i^{(1)}(\mathbf{x}')\right)$$

   From here, we recognize the multiplicand to be a valid kernel, using our result from part (a). Then, recognize that a product of valid kernels is a valid kernel, per our results in part 4. Alternatively, write

$$K(\mathbf{x}, \mathbf{x}') = \exp(K^{(1)}(\mathbf{x}, \mathbf{x}')) = \lim_{i\to\infty}\left(1 + \boldsymbol{\phi}^{(1)}(\mathbf{x})^\top \boldsymbol{\phi}^{(1)}(\mathbf{x}') + \cdots + \frac{(\boldsymbol{\phi}^{(1)}(\mathbf{x})^\top \boldsymbol{\phi}^{(1)}(\mathbf{x}'))^i}{i!}\right)$$

   and use properties 1, 2, and 4 to recognize that this is a valid kernel.

   Check 2.5 You found an infinite basis to represent the exponential function as a dot product.

   Check 2.6 You correctly proved that the kernel is valid, not necessarily using the result from part (a).

6.
- $K_0(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$ is a valid kernel by definition of the kernel (it is the inner product of $\mathbf{x}$ and $\mathbf{x}'$).
- Thus $K_1(\mathbf{x}, \mathbf{x}') = \exp(2\mathbf{x}^\top \mathbf{x}')$ is also a valid kernel by property 1 and 5.
- Note that $K(\mathbf{x}, \mathbf{x}') = \exp(-\mathbf{x}^\top \mathbf{x}) \exp(2\mathbf{x}^\top \mathbf{x}') \exp(-\mathbf{x}'^\top \mathbf{x}') = f(\mathbf{x}) K_1 f(\mathbf{x}')$, where $f(\mathbf{x}) = \exp(-\mathbf{x}^\top \mathbf{x})$.
- Hence using property 3 in the last step, we proved $K(\mathbf{x}, \mathbf{x}')$ is a kernel.

Note: a common mistake is saying $\exp(-\mathbf{x}^\top \mathbf{x})$ is a kernel. It is not.

Check 2.7 You specified a series of steps using the various properties to show that it is a valid kernel.

End Solution

Problem 3 (Scaling up your SVM solver, 10pts (+opportunity for extra credit))

For this problem you will build a simple SVM classifier for a binary classification problem. We have provided you two files for experimentation: training *data.csv* and validation *val.csv*.

- First read the paper at http://www.jmlr.org/papers/volume6/bordes05a/bordes05a.pdf and implement the Kernel Perceptron algorithm and the Budget Kernel Perceptron algorithm. Aim to make the optimization as fast as possible. Implement this algorithm in *problem3.py*.

  [Hint: For this problem, efficiency will be an issue. Instead of directly implementing this algorithm using numpy matrices, you should utilize Python dictionaries to represent sparse matrices. This will be necessary to have the algorithm run in a reasonable amount of time.]

- Next experiment with the hyperparameters for each of these models. Try seeing if you can identify some patterns by changing β, N (the maximum number of support vectors), or the number of random training samples taken during the Randomized Search procedure (Section 4.3). Note the training time, training and validation accuracy, and number of support vectors for various setups.

- Lastly, compare the classification to the naive SVM imported from scikit-learn by reporting accuracy on the provided validation data. *For extra credit, implement the SMO algorithm and implement the LASVM process and do the same as above.*[a]

We are intentionally leaving this problem open-ended to allow for experimentation, and so we will be looking for your thought process and not a particular graph. Visualizations should be generated using the provided code. You can use the trivial $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$ kernel for this problem, though you are welcome to experiment with more interesting kernels too.

In addition, provide answers the following reading questions **in one or two sentences for each**.

1. In one short sentence, state the main purpose of the paper.
2. Describe each of the parameters in Eq. 1 in the paper
3. State, informally, one guarantee about the Kernel perceptron algorithm described in the paper.
4. What is the main way the budget kernel perceptron algorithm tries to improve on the perceptron algorithm?
5. (*if you did the extra credit*) In simple words, what is the theoretical guarantee of LASVM algorithm? How does it compare to its practical performance?

[a]Extra credit only makes a difference to your grade at the end of the semester if you are on a grade boundary.

Solution

Reading questions

1. The goal of the paper is to achieve approximations of the SVM QP solution at a lower computational cost to handle larger datasets.

   Check 3.1 Your answer closely matches the one above.

2. The parameters are:
   - $\hat{y}$: value to be thresholded to get prediction.
   - $\boldsymbol{\phi}(\mathbf{x})$: feature vector for the data.
   - $\mathbf{w}$: weight vector. Is a parameter of the model.
   - b: bias term. Is a parameter of the model.

Check 3.2 Your answer identifies each of the terms above.

3. One guarantee of the Kernel perceptron algorithm is that it converges after a finite number of misclassifications/insertions of support vectors (cf. Novikoff's Theorem) if a solution exists (the mapping is linearly separable).

   Check 3.3 Your answer closely matches the one above.

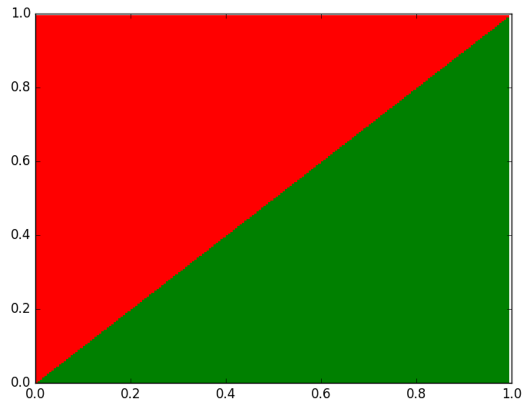
4. The main way the budget kernel perceptron improves on the perceptron algorithm is to allow for the removal of support vectors in order to reduce overfitting while increasing the margin.

   Check 3.4 Your answer closely matches the one above.

5. The theoretical guarantee of the LaSVM algorithm is that it converges to the SVM QP solution after sufficiently many passes over the data. In practice, the LaSVM algorithm outputs a good approximation after 1 pass over the data.

   Check 3.5 Your answer closely matches the one above.

Programming questions Your image for the correct implementation of KernelPerceptron and BudgetKernelPerceptron should look like



Check 3.6 Your image looks like the above. It is fine for your decision boundary to be jagged (i.e. looks like a step function), as long as the direction and colors are correct.

Next, you were asked to some experimentation to see how changing the parameters affects the outcome. You should have done at least two of the following:

- adjust β
- adjust N
- adjust number of samples
- changed the Kernel

Your analysis should focus on (at least) changes to accuracy and speed. $\beta = 0$ would produce a valid decision boundary. Positive values of β would lead to adjustments if $y * \hat{y}$ for a point was negative (meaning the point was misclassified) or within β of 0 (meaning not correctly classified by enough). Similarly, negative β means that you are okay with occasionally being wrong about a classification, as long as $\hat{y}$ is within β of 0. These should, respectively, have slowed down and sped up your algorithm, relative to $\beta = 0$.

Changing N changes the maximum number of support vectors. You will likely find that very low values of N like 10 are quite fast per epoch (and overall), but not as highly performant. You likely will not get a correct decision boundary with this. With larger N, you get less speed performance, since you have more support vectors, but better accuracy.

Increasing the number of samples likely slowed down your algorithm since it has to iterate through more samples. However, having too few will decrease accuracy since you may not have enough training data (you may not have enough support vectors).

Changing the Kernel could have varied behavior, and you are expected to explain what happened and why you think it did.

Check 3.7 You adjusted at least one of the above parameters and noted what happened, which should be close to what is described above.

Check 3.8 You compared the classification to the naive SVM imported from scikit-learn by reporting accuracy on the provided validation data.

End Solution

Calibration [1pt]

Approximately how long did this homework take you to complete?