Final, CS-GY 6923

Sample Questions

Show all of your work to receive full (and partial) credit.

Always, Sometimes, Never

Indicate whether each of the following statements is **always** true, **sometimes** true, or **never** true. Provide a one or two short justification or example to explain your choice.

1. Given a linearly separable data set, an optimal solution to the soft-margin SVM objective will be a correct separating hyperplane for the dataset.

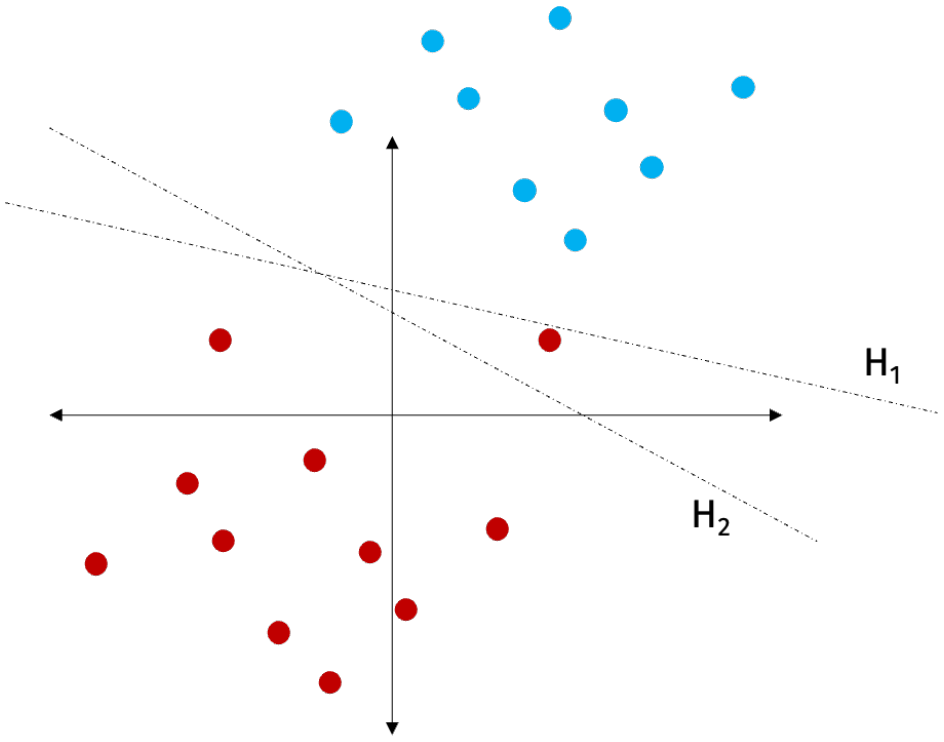
   ALWAYS SOMETIMES NEVER

2. Let K be a kernel gram matrix generated from a datasets $x_1, \ldots, x_n$ and a PSD kernel function k. K has a symmetric Signular Value Decomposition (meaning the same left and right singular vectors).

   ALWAYS SOMETIMES NEVER

Short Answer

3. In the plot below, H_1 and H_2 are hyperplanes obtained by trainging a soft-margin SVM with different values of C. Which one was trained with a larger value of C? On the same plot draw the hyperplane that you believe would be returned by a hard margin SVM.



4. TRUE or FALSE. A convolutional layer that takes in $n \times n$ images and processes them with a series of $n \times n$ convolutional filters is equivalent to a fully connected layer.
5. An alternative definition of a PSD kernel function that you will see in many text books is as follows: We say that k is PSD if for *any* dataset $x_1, \ldots, x_n$, the kernel gram matrix K with $K_{ij} = k(x_i, x_j)$ is ``positive semi-definite'', where we say a matrix K is positive semidefinite if, for *all* vectors x, $x^T K x \geq 0$ (this is a definition you might have seen before in a linear algebra class).

   Prove that the other definition we gave for k (i.e. that $k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ for some feature transformtion ϕ Implies this definition (you don't need to show the other way).
6. Draw a linearly seperable data set where there is no unique maximum margin classifier. I.e. a hard margin SVM could return multiple different solutions.
7. As discussed in class, convolutional networks can be viewed as special cases of fully connected networks with weight sharing. Of course, weight sharing effects how gradients of computed for the neural network. Revisit Problem 3(d) on Homework 3 and recommute the gradient for the network where we have the additional weight sharing constraints that $W_{H,2} = W_{H,3}$ and $W_{O,1} = W_{O,2}$.