# Homework 4 – Machine Learning (CS4342, Whitehill, Spring 2023)

You may do this homework either by yourself or in a team of 2 people.

1. **Support Vector Machines: Training [45 points]**: In this problem you will implement a class called `SVM4342` that supports both training and testing of a linear, hard-margin support vector machine (SVM). In particular, you should flesh out the two methods `fit` and `predict` that have the same API as the other machine learning tools in the `sklearn` package.

   (a) `fit`: Given a matrix $\mathbf{X}$ consisting of $n$ rows (examples) by $m$ columns (features)[1] as well as a vector $\mathbf{y} \in \{-1, 1\}^n$ consisting of labels, optimize the hyperplane normal vector $\mathbf{w}$ and bias term $b$ to maximize the margin between the two classes, subject to the constraints that each data point be on the correct side of the hyperplane (hard-margin SVM). To do so, you should harness an off-the-shelf quadratic programming (QP) solver from the `cvxopt` Python package (it is probably easiest just to use Google Colab at `https://colab.research.google.com/`, but if you want to install it on your own machine then see `https://cvxopt.org/install/`). The template file already includes code showing how to use it – just pass in the relevant matrices and vectors as `numpy` arrays. The trick is in setting up the variables to encode the SVM **objective** and **constraints** correctly, as described in lecture. Then, store the optimized values in `self.w` and `self.b` because you'll need them later.

   (b) `predict`: Given a matrix $\mathbf{X}$ of test examples, and given the pre-trained `self.w` and `self.b`, compute and return the corresponding test labels.

   Your code will be evaluated based both on the accuracy of the predicted test labels in several tests (you can see the exact test code we'll be running in the template file), as well as the difference between the hyperplane parameters compared to their ideal values.

   Note: as shown in the template, you can calculate the ideal values using the off-the-shelf `sklearn` SVM solver. Note that, since the SVM in this package is, by default, a soft-margin SVM, we have to "force" it to be hard-margin by making the cost penalty $C$ very large.

   See the starter code `homework4_template.py`.

2. **Support Vector Machines: Plotting the Hyperplane [10 points]**: Finish the code inside `test1` in `homework4_template.py` so that it plots the optimal separating hyperplane $H$ obtained from your SVM implementation on top of the scatter plot of the data points themselves. You will need to convert from the standard ($\mathbf{x}^\top \mathbf{w} + b = 0$) form of the hyperplane into a more convenient form to compute `yvals`. Put the plot into a PDF file and submit it on Canvas.

Combine both your Python file(s) and the PDF in a single Zip file, and then submit the Zip file on Canvas. If you are working as part of a team, then make sure you register as one of the pre-allocated teams on Canvas; only one person should then submit the Zip file (and you will both receive credit).

[1] To be consistent with the notation I've used in the lecture slides, $\mathbf{X}$ in this assignment would actually need to be called $\mathbf{X}^\top$. The reason I call it $\mathbf{X}$ here is to be consistent with the `sklearn` package.