550.740 Project 4; Support Vector Machines.
Due on Wednesday May 1st (last day of class).

- **Please type your answers (using, e.g., LateX or MS Word.)**
- *The solution must be written with sufficient explanations and your results must be commented. Returning a series of numerical results and figures is not enough. A solution in the form of a program output is not acceptable either.*
- *Please return your program sources. They will not graded (so no direct credit for them), but they will be useful in order to understand why results are not correct (and decide whether partial credit can be given) and to ensure that your work is original. You may use any programming language, although Python, Matlab or R are recommended.*
- *The ".csv" files associated with this project are available on Blackboard.*

Preparation

This project studies support vector machines for classification. For this purpose, you can use any SVM implementation that offers the following features.

- Given input data $(x_1, y_1), \ldots, (x_N, y_N)$, it must be able to solve the minimization problem in dual space with objective function

$$\frac{1}{2}\sum_{k,l=1}^{N} \alpha_k \alpha_l y_k y_l K(x_k, x_l) - \sum_{k=1}^{N} \alpha_k$$

  under the constraints $0 \leq \alpha_k \leq \gamma$, $\sum_{k=1}^{N} \alpha_k y_k = 0$.
- The program should allow you to take $\gamma = 1$ and to specify your own function evaluating the positive kernel K

The SVC class in SciKit-learn, the function fitcsvm in Matlab and the package ksvm in R seem to satisfy these requirements (I only tested the first one).

Part 2 of the project studies Support Vector Machines for classification on the USPS digit database. This dataset is available on blackboard in three parts: training, validation and testing. The use of the different parts will be explained in the questions. In these files, the class information is, as usual, in the last column.

Question 1.

You will use in this question a Cauchy kernel defined by

$$K_\sigma(x, x') = \frac{1}{1 + \frac{|x-x'|^2}{\sigma^2}}.$$

Use the training set stored in *kernel1Train.csv*, in which each row contains the two coordinates of a training sample followed by its class. The test set, stored in *kernel1Test.csv*, contains point regularly spaced on a 100x100 2D grid, with associated class variable.

(1) Use a program that reshapes the test data from 10,000x1 to 100x100 and visualizes it as an image (and show the image as your answer to this question).

(2) Train an SVM on the training data for $\sigma = \{0.01, 0.025, 0.05, 0.1, 0.5, 1.0, 2.5\}$. For each value of σ, report the classifier's error (in average) on the test set, and provide an image similar to that built in the previous question in which the true classes are replaced by the predicted ones.

Question 2.

You will have to build several binary SVM's in this question, using a subset of the pendigits data set. This subset is divided in three parts: pendigitsTrain.csv, pendigitsValid.csv and pendigitsTest.csv, for training, validation and testing.

The first two datasets will be used for training including the computation of an "optimal" value for σ. This optimal σ is computed by minimizing the prediction error (over the validation set) of a classifier estimated (from the training set) using σ. For this project, the optimization will only be over four values of $\sigma \in \{1., 5., 10., 50., 100., 150., 200.\}$. The complete training algorithm will therefore first run, for each σ:

(1) Use pendigitsTrain to estimate an SVM with the kernel K_σ.
(2) Compute the classification error on the pendigitsValid $\varepsilon(\sigma)$.

The algorithm will then select the value of σ for which ε is smallest and relearn an SVM for this value on the union of the training set and of the validation set. The test set is, of course, not used during this process.

The class values for pendigits are $\mathcal{G} = \{0, 1, \ldots, 9\}$. Each sample is a sequence of 8 points in 2D (16 values) extracted from a handwritten digit. The general approach proposed in the three questions below requires on the following construction. If v is a binary (0 or 1) vector of size 10 (the number of classes), one can train (following the procedure that was just described) a binary SVM, denoted f_v, with values in $\{0, 1\}$, in order to distinguish whether a sample belongs to a class g for which $v(g) = 1$ or in a class for which $v(g) = 0$. Given a family $v_1, \ldots, v_m$ (for some integer m), one then computes the final classifier, which associates to all inputs x

$$\varphi(x, g) = \sum_{j=1}^{m} \chi_{v_j(g) = f_{v_j}(x)}$$

and

$$f(x) = \mathrm{argmax}_g \varphi(x, g),$$

i.e., each classifier f_{v_j} votes for a class and the decision is the class that collects the most votes. In case of tie, let your classifier return a class at random among the top ones.

(1) Using $K_\sigma(x, x') = \left(1 + \frac{|x-x'|}{\sigma} + \frac{|x-x'|^2}{3\sigma^2}\right) e^{-\frac{|x-x'|}{\sigma}}$, compute the previous estimator with $m = 10$ and $v_1, \ldots, v_{10}$ correspond to the m columns of the ten-dimensional identity matrix. Run it on the test set and provide the resulting "confusion matrix",

whose entries $a(g, g_0), g, g_0 \in \mathcal{G}$ provides the fraction of testing examples whose true class is g_0 that are predicted to belong in class g.

(2) Answer the same question with $m = 20$ and $v_1, \ldots, v_m$ given by random permutations of v_0 containing five zeros and five ones (making sure that no pair of them coincides).

(3) Same question with K_σ given as in Question 1.