In this assignment we continue our investigation of the "Pima" dataset. As in the previous assignment, you can download the dataset (pima.txt) and its description (pima_desc.txt) from the course web page. In addition to the complete dataset pima.txt, you have pima_train.txt and pima_test.txt you will need to use for training and testing purposes. The dataset has been obtained from the UC Irvine machine learning repository:

Problem 1. Support vector machines

Support vector machines represent yet another technique one can apply to the problem of binary classification. The idea is to find the hyperplane that separates the examples in two classes the best. The best hyperplane is defined in terms of the maximum margin. The learning problem reduces as usually to optimization, in this case, a quadratic optimization problem.

There is a number of implementations of SVM algorithms with better or worse running time performances. Here we use a Matlab code implementing SVM solver for the linear decision boundary proposed by O.L. Mangasarian and D. Musicant. The paper describing this method can be downloaded electronically at:
http://www.ai.mit.edu/projects/jmlr/papers/volume1/mangasarian01a/html/. The SVM solver is in files svml.m and svml Itsol.m that can be downloaded from the course web page. svml Itsol.m is a slightly modified version of the original program by O.L. Mangasarian and D. Musicant. To run it you call svml.m that takes care of converting outputs from 0,1 class labels to -1,1 (!!) and sets other parameters of the Lagrangian SVM.

Part a. Use the svml code to learn the weights w and b (bias) of the linear model on the training set. Assume the cost for crossing the boundary (a parameter of the svml procedure) is 1.

Part b. Write and submit a function apply_svlm(x, w, b) that takes an input vector x, and parameters w, and b of the liner SVM and outputs the class decision (use 0 and 1) for the input x. Briefly, the class is 1 if \( w^T x + b \geq 0 \), and 0 otherwise.
Part c. Use function `apply_svm(x, w, b)` to calculate the confusion matrices and other stats for both the training and testing data. More specifically, please report the confusion matrix, misclassification error, sensitivity and specificity for both the training and testing set.

Part d. Compare the above results to the results from Homework assignment 5 obtained for the logistic regression and the Naive Bayes models. Which model do you think performed the best?

Part e. Use `percurve` to plot the ROC curve and calculate AUC for the SVM model on the testing set. To do so, for each data example $x$ in the testing data, calculate the score $\text{score}(x) = w^T x + b$. Compare the AUC for the SVM with the AUCs for the logistic regression and for the Naive Bayes in homework assignment 5. Which model is better?

**Optional.** If you are interested experimenting with existing SVMs tools including tools supporting non-linear kernels please check out the following software packages: liblinear, libsvm and svmlight. All these can be interfaced with Matlab.