HW Assignment 2 (Due date: October 10, Monday)

1 Problems

1. [**Fisher Criterion and Least Squares**, 30 points]
   Exercise 4.6, page 221.

2. [**Perceptrons**, 50 points]
   Consider a training set that contains the following 8 examples:

$\phi_1(x)$	$\phi_2(x)$	$\phi_3(x)$	$t(x)$
0	0	0	+1
0	1	0	+1
1.5	0	-1.5	+1
1.5	1	-1.5	+1
1.5	0	0	–1
1.5	1	0	–1
0	0	-1.5	–1
0	1	-1.5	–1

   (a) Prove that the perceptron algorithm does not converge on this dataset.

   (b) Consider a kernel perceptron that uses a polynomial kernel $k(x, y) = (1+\phi(x)^T\phi(y))^d$. What is the smallest degree d for which the kernel perceptron would converge on this dataset?

3. [**Normalized Kernels**, 75 points]
   Let $K : X \times X \to R$ be a kernel function defined over a sample space X.

   (a) Prove that the function below (a *normalized kernel*) is a valid kernel.

$$\frac{K(x, y)}{\sqrt{K(x, x)K(y, y)}}$$

   (b) Which types of input data would benefit from normalizing the kernel function? Explain why, and provide real world examples.

   (c) Why it would not make sense to normalize a Gaussian kernel?

4. [**Support Vector Machines**, 50 points]
   Prove that the sum of slacks $\sum \xi_n$ from the objective function of the SVM formulation with soft margin is an upper bound on the number of misclassified training examples.

5. [**Digit Recognition**, 250 points]
In this exercise, you are asked to run an experimental evaluation of SVMs and the perceptron algorithm, with and without kernels, on the problem of classifying images representing digits.

   (a) Implement the 4 versions of the perceptron algorithm discussed in class: perceptron, averaged perceptron, kernel perceptron, and averaged kernel perceptron. The algorithms should stop after achieving convergence, or after a predefined number of epochs T, whichever comes first. The kernel versions should contain a call to a kernel function that is implemented separately. In general, your code should be modular and as efficient as possible.

   (b) Validate experimentally the conclusions you reached for Exercise 2.

   (c) The UCI Machine Learning Repository at *www.ics.uci.edu/˜mlearn* maintains datasets for a wide variety of machine learning problems. For this assignment, you are supposed to work with the Optical Recognition of Handwritten Digits Data Set. The webpage for this dataset is at:
   http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits
   The actual dataset is located at:
   http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/

   Read the description of the dataset. Download the training set `optdigits.tra` and the test set `optdigits.tes`. Use the first 1000 examples in `optdigits.tra` for *development* and the rest of 2823 examples for *training*. Use all 1797 examples in `optdigits.tes` for *testing*. Create training files for each of the 10 digits, setting the class to 1 for instances of that digit, and to -1 for instances of other digits, i.e. *one-vs-rest* scenario. Scale all the features to be in the interval [0, 1].

   (d) Train first the linear perceptron, with the number of epochs set to $T \in \{1, 2, ..., 20\}$. After training the linear perceptron, normalize the learned weight vector. Select for T the value that obtains the best overall accuracy on the development data, and use this value for the remaining experiments.

   Run experiments with the four versions of the perceptron algorithm. For the kernel perceptron, experiment with polynomial kernels $k(\mathbf{x}, \mathbf{y}) = (1+\mathbf{x}^T\mathbf{y})^d$ with degrees $d \in \{2, 3, 4, 5, 6\}$, and with Gaussian kernels $k(\mathbf{x}, \mathbf{y}) = exp(-||\mathbf{x}-\mathbf{y}||^2/2\sigma^2)$ with the width $\sigma \in \{0.1, 0.5, 2, 5, 10\}$. For each hyper-parameter value, you will have trained 10 models, one for each digit. In order to compute the label for a test or development example, you will run the 10 trained models and output the label that obtains the highest score. Compute the overall accuracy on the development data and identify the hyper-parameter value that obtains the best accuracy. Use the tuned hyper-parameter (d for poly-kernel, σ for Gaussian) to compute the overall performance on the test data.

   For each version of the perceptron report the total training time, the overall accuracy, and the number of support vectors. Show and compare the corresponding 4 confusion matrices. Which digit seems to be the hardest to classify? Which perceptron/kernel combination achieves the best performance? Which algorithms are slower at training time, and why?

(e) Run the same experiments using SVMs instead of perceptrons, i.e. linear SVMs and SVMs with polynomial and Gaussian kernels. Use the same tuning scenarios for the hyper-parameters of the polynomial and Gaussian kernels. Use $C = 1$ in all SVM experiments. Report the same types of results and analysis as above, and compare with the perceptron results.

2 Tools

You are free to use MATLAB, R, or packages written in C++/Java/Python to complete the SVM part of this assignment. I recommend using SVMLIGHT, a C implementation, or LIBSVM which has both Java and C++ implementations. Their web sites contain plenty of documentation on how to use them.

3 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit a directory that has your working code, any trace files, and a concise README file describing these before class. Create a gzipped, tar ball archive of your directory, and upload it on Blackboard by the due date.

For example, if the name is John Williams, creating the archive can be done using the following commands:

> tar cvf williams_john.tar williams_john

> gzip williams_john.tar

These two steps will create the file 'williams_john.tar.gz' that you can upload on Blackboard.

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.
2. Use adequate comments, both block and in-line to document your code.
3. **Do not submit third-party ML packages on Blackboard!** Just explain in the REAMDE file how you use external packages.
4. Type and nicely format the project report, including discussion points, tables, graphs etc. so that it is presentable and easy to read.
5. Working code and/or correct answers is only one part of the assignment. The project report, including discussion of the specific issues which the assignment asks about, is also a very important part of the assignment. Take the time and space to make an adequate and clear project report. On the non-programming learning-theory assignment, clear and complete explanations and proofs of your results are as important as getting the right answer.