Homework 5: SVM

This homework is based on the material in http://www.idiap.ch/~fleuret/files/EE613/EE613-pw12.pdf. You are supposed to implement this homework in python or R. A skeleton of the answers is given in octave.

The goal of this lab session is to experiment with SVM for binary classification, and in particular to understand the role of the hyperparameters.

Exercise 1

Question a. In this question, the goal is to apply a linear SVM to two-dimensional data points.

1. **Load and visualize the data**. Run ex1a.m. Visualize the data. Do you think the classes are linearly separable?

2. **Learn the SVM**. Use MATLAB function *svmtrain* to learn a linear SVM on these data points. Run help *svmtrain* for the documentation. In particular, experiment with different values of the cost parameter C, e.g. trying for C = 1, C = 100 or more. You can visualize the learned model with the 'ShowPlot' option set to true. Do you observe any differences in the learned hyperplane for different values of C? In the evolution of the support vectors? Comment on your observations.

3. **Linear kernel**. Implement the linear kernel function $K(x_i, x_j) = x_i^T \cdot x_j$ in linear_*kernel.m*.

4. **Estimate the decision boundary**. The function *svmtrain* with the 'ShowPlot' option set to true displays the decision boundary with a black line. In the current case, this boundary is given by a line. First remind the expression providing the weights and bias of the decision line. Second, compute these weights and bias, and plot the decision line. Get the important information from the learned model: model.SupportVectors, model.Alpha, model.Bias. **Warning**: In MATLAB, model.Alpha contains the weights already multiplied by the labels, i.e. $\alpha_i \times y_i$. Verify that the line you obtain matches the line visualized with *svmtrain*.

Question b. In this question, the goal is to apply a linear SVM to classify emails as spam/nonspam.

1. **Load the data**. Run ex1b.m. The data are organized in matrices, where each row represents one document (an email), and each of the 2500 columns represents one particular word in a dictionary. For a particular row (document), the value at column j is the number of times the j^{th} word of the dictionary has occurred in the document. When loading, the data are separated into two sets: the training set and the test set.

2. **Learn and test SVM models**. Use *svmtrain* and *svmclassify* to learn SVM models on the training set and apply them on the test data. Visualize the trained weights, and how they evolve as you change the number of number of training samples (you can use $numTrainDocs = 50, 100, 400, 700$).

3. **Compute the accuracy**. Compute the classification accuracy given the predicted labels obtained with *svmclassify* vs. the true labels test labels of the test data. Change the number of training documents *numTrainDocs*, and comment on the effect of the number of training samples on the accuracy.

Exercise 2

Question a. In this question, the goal is to apply an SVM with an RBF kernel to two-dimensional data points.

1. **Load and visualize the data**. Run ex2a.m. Visualize the data.
2. **Learn the SVM**. Use MATLAB function *svmtrain* to learn an SVM with an RBF kernel on these data points. Set parameter values to $C = 1$, $\gamma = 100$. Remember $\gamma = \frac{1}{2\sigma^2}$ You can visualize the learned model with the '*ShowPlot*' option set to true.
3. RBF kernel. Implement the RBF kernel function $K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2)$ in *rbfkernel.m*.
4. **Visualizing the decision cost**. The goal is to visualize the decision cost output, as well as determine the decision boundary as the zero-contour of the SVM score. We recall that the classification score $f(x)$ for any data point x can be expressed as:

$$f(x) = \sum_i \alpha_i y_i K(x_i, x) + b \qquad (1)$$

where $\{x_i\}$ are the support vectors, $\{y_i\}$ their labels, $\{\alpha_i\}$ their weights, K is the kernel function and b the bias.

To visualize the cost, a grid is defined over the data space. For each grid location indexes (i, j), compute the classification score $Z(i, j)$. For that purpose, complete *rbf scoring function.m* using Equation 1 and your kernel function defined in Question a.3. Visualize both the cost surface, as well as the zero-contour (and other constant cost contour lines) of the SVM score over the grid. For the contour, you can use *c=contour(X,Y,Z,[0 0],'color','k')* which will plot the zero-contour of the SVM score over the grid. Verify that this contour corresponds to the decision boundary observed in Question a.2.

Question b. In this question, the goal is to visualize the effect of hyperparameters C and γ of the SVM with RBF kernel on another two-dimensional example.

1. **Load and visualize the data**. Run ex2b.m. Visualize the data. Do you think the classes are easily separable?
2. **Learn SVM models for different hyperparameter values**. Try with $C = 1, 1000$ and $\gamma = 1, 10, 100, 1000$. Visualize the learned models with different combinations of hyperparameter values. Have a look as well at the number of support vectors in the different configurations. Comment on your observations.

Question c. In this question, the goal is to perform cross-validation to find the best hyperparameters.

1. **Load the data**. Run ex2c.m. The data points of Question b. have been split into 3 sets V1, V2, V3 for cross-validation. A search grid is defined over the parameter space.

2. **Perform cross-validation**. Follow the code that has already been implemented. Explain in a few words the process of 3-fold cross-validation that is applied on this example. Update missing information in the call to function *svmtrain*, and complete the line where classification accuracy is performed. Report the best set of parameters (C, γ) that has been returned by the grid search. Visualize again the model of Question b.2 learned with these parameters. What can you conclude?