Technische Universität München — **WS 2015**
Fakultät für Informatik — **Exercise Sheet 2**
Prof. N. Navab, Ph.D. — **Nov 18th, 2015**

Practical Course: Machine Learning in Medical Imaging

Linear Classifiers and SVM

1 Kernel Ridge Regression

Ridge regression is an extension to ordinary least squares by adding a regularization term to the loss function. It is defined as

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2 + \lambda \|\boldsymbol{\beta}\|_2^2, \tag{1}$$

where the value of $\lambda > 0$ determines the amount of regularization. By replacing $\boldsymbol{\beta}$ with $\sum_{i=1}^{n} \alpha_i \mathbf{x}_i$ we obtain

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n} \alpha_j \mathbf{x}_i^T \mathbf{x}_j \right)^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j \tag{2}$$

As in support vector machines, we can use the Kernel trick to make ridge regression non-linear and at the same time avoid explicitly transforming features. By specifying $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$, we obtain the objective function of Kernel Ridge Regression:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n} \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \right)^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{3}$$

Estimation of $\boldsymbol{\alpha}$ can still be achieved in closed form with $\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$, where $\mathbf{K}$ is the Kernel matrix with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The decision function then becomes

$$f(\mathbf{x}_0) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_0). \tag{4}$$

Tasks

1. Create a function `kernel_ridge_fit`, which takes a matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ of samples, a vector $\mathbf{y} \in \mathbb{R}^n$ of outcomes for each sample, and a *kernel function*. `kernel_ridge_fit` should return the estimated vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ and the *mean squared error* on the training samples.
2. Create a function `kernel_ridge_predict`, which takes the matrix $\mathbf{X}$ used during training, the vector $\boldsymbol{\alpha}$, a matrix $\mathbf{X}_t$ of samples for testing and their respective outcomes $\mathbf{y}_t$, and a *kernel function*. The function should return a vector of predicted outcomes and the *mean squared error* on the testing samples.

3. Apply your implementation of kernel ridge regression to the two datasets:

   `two-moons.mat` This dataset is already split in training and testing data.

   `Aqua-all.csv` Here, the first column denotes the outcome **`y`**, the remaining columns the features. Use the first 100 samples as training data for the `kernel_ridge_fit` function. Use the remaining 97 samples for testing in `kernel_ridge_predict`.

   a) Plot the mean squared *training error* of the training samples obtained from `kernel_ridge_fit` for

      - $\lambda \in \{0.001, 0.01, 0.1, 1, 10, 20, 50, 100, 200, 500, 1000, 10000\}$
      - the following Kernel functions:
        - linear,
        - polynomial ($c = 1$, $d = 2$),
        - sigmoid ($\gamma = -0.001$, $c = 1$),
        - RBF ($\gamma = 0.001$).

      The x axis should denote the λ values in log scale and the y axis the mean squared error.

   b) Use the testing data to evaluate all models trained above and plot the mean squared *test error* obtained from `kernel_ridge_predict`. The x-axis should denote the λ values in log scale and the y-axis the mean squared error.

   c) For the `two-moons` dataset visualize also 2D plots comparing the predicted vs true errors.

2 SVMs

Repeat the exercise above by using either the MATLAB embedded SVM library or by installing and downloading `libsvm`. Kernel Ridge Regression does not promote the sparseness i.e. there is no concept of support vectors as in SVMs. Compare the values of the alpha vectors in each case.