Experiment 5: SVM

November 9, 2018

1 Description

This exercise gives you practice with using SVMs for both linear and non-linear classification.

2 SVM

The fist part is to implement a regularized SVM classifier. Its details can be found in the lecture slides. We hereby only give a sketch about SVM. The regularized SVM can be formulated as

$$\begin{aligned} \min_{\omega,b,\xi} \quad & \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y^{(i)}\left(\omega^T x^{(i)} + b\right) \geq 1 - \xi_i, \ \forall i = 1, \cdots, m \\ & \xi_i \geq 0, \quad \forall i = 1, \cdots, m \end{aligned}$$

where ξ_i is a slack variable. It dual problem can be defined as

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j < x^{(i)}, x^{(j)} > \\ s.t. \quad & 0 \leq \alpha_i \leq C, \quad \forall i = 1, \cdots, m \\ & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \end{aligned}$$

We can get a SVM classifier by solving the above QP problem. For simplicity purpose, we can utilize exiting QP solvers (e.g., quadprog in MATLAB or qp in Octave).

Two data sets are given, each of which are divided into two parts for training and testing, respectively. In particular, the first data set includes traing_1.txt for training and test_1.txt for test, and the second one includes traing_2.txt and test_2.txt In these data files, the first two columns are features, and the last one is label. Try the SVM on the two datasets, respectively, and **answer the following questions**

1. Plot the decision boundary of the SVM with the training data.
2. Use the test data to evaluate the SVM classifier and show the fraction of test examples which were misclassified

3. Try different values of the regularization term C, and report your observations.

3 Handwritten Digit Recognition

The second part is to apply your SVM classifier to recognize handwritten digits. Data sets for training and testing have been given in train-01-image.svm and test-01-images.svm. To simplify things, we will only be distinguishing between 0's and 1's. The training data set includes 12665 images while the one for testing contains 2115 images. Each row represents an image, where the first term is the label, while the follows are indices of the pixels and the corresponding gray values. Note that only the pixels with non-zero gray values are given. **Please carefully read and try strimage.m where how to process the data examples is given. Show some data examples in your report.** Train your SVM model by the training data and apply it to recognize the handwritten digits given in the test data set. **Answer the following questions**

1. Train a plain-vanilla SVM (i.e, no regularization with $\xi_i = 0$ for $\forall i$) on train-01-images.svm. What is the training error, i.e., the fraction of examples in the training data that are misclassified? From the set of misclassified images, pick one. Attach of plot of it with your solution. Why do you think the SVM fails to classify it correctly during training? Now apply the SVM to the test set test-01-image.svm; what is the test error, i.e., the fraction of test data that is misclassified?

2. Experiment with different values of the regularization term C. Start by guessing/estimating a range in which you think C should lie. Then choose the values of C (within that range) at which you will evaluate the performance of the SVM. You need not pick more than 10 such values, though you should feel free to pick as many (or as few!) as you want. For these values of C, plot the corresponding error. (i) What value of C gives you the best training error on the dataset from part? (ii) How does the test error for this choice of C compare with the test error you computed in part (i)? (iii) Could you optimize C so that it leads to the best test error, rather than the training error? Should you?

4 Non-Linear SVM

One way to enable non-linear SVM is to introduce kernels. Recall that linearly non-separable features often become linearly separable after they are mapped to a high dimensional feature space. However, we do not ever need to compute the feature mappings $\phi(x^{(i)})$ explicitly: we only need to work with their kernels, which are easier to compute. Therefore, it's possible to create a very complex decision boundary based on a high dimensional (even infinite dimensional) feature mapping but still have an efficient computation because of the kernel representation.

In this exercise, you will apply Radial Basis Function (RBF) kernel (instead of regularization approach) to SVM model. This kernel has the formula

$$K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)}) = \exp\left(-\gamma \|x^{(i)} - x^{(j)}\|^2\right), \quad \gamma > 0$$

Notice that this is the same as the Gaussian kernel, except that term $\frac{1}{2\sigma^2}$ in the Gaussian kernel has been replaced by γ. Once again, remember that at no point will you need to calculate $\phi(x)$ directly. In fact, $\phi(x)$ is infinite dimensional for this kernel, so storing it in memory would be impossible.

Now let's see how an RBF kernel can choose a non-linear decision boundary. Load the data set training_3.txt into your Matlab/Octave workspace. This is a two-dimensional classification problem. **Plot the positives and negatives using different colors. Does there exist a linear decision boundary for this dataset?**

Train an SVM model on an RBF kernel with $\gamma = 100$ according to the above exercises. Once you have the model, you need to **visualize the decision boundary**, referring to the following codes

```
% Make classification predictions over a grid of values
xplot = linspace(min(features(:,1)), max(features(:,1)), 100)';
yplot = linspace(min(features(:,2)), max(features(:,2)), 100)';
[X, Y] = meshgrid(xplot, yplot);
vals = zeros(size(X));

% For each point in this grid, you need to compute its decision
% value. Store the decision values in vals.
... ...

% Plot the SVM boundary
colormap bone

contour(X,Y, vals, [0 0], 'LineWidth', 4);
```

Recall that, it is function $\sum_i \alpha_i K(x^{(i)}, x) + b$ giving the decision values that are used to make a classification. An example x is classified as a positive example if $\sum_i \alpha_i K(x^{(i)}, x) + b \geq 0$, and is classified negative otherwise.

Now train your model using γ values of 1, 10, 100, and 1000 and plot the decision boundary (using no contour fill) for each model. **How does the fit of the boundary change with γ?**

5 Sequential Minimal Optimization

This is an optional excise. You can get extra credits, if you could use *Sequential Minimal Optimization* (SMO) algorithm to replace the standard QP solvers.