Machine Learning (CS 6140)

Assigenment 2

Statistics, Perceptron, Kernels, SVM, Multi-Layer Perceptron

Instructor: Professor Lu Wang

Due Date: March 2nd, 2017, beginning of class

For the programming questions, you can use Java, Python, C/C++, or R. You also need to include a README file with detailed instructions on how to run the code and comments to facilitate understanding. No need to print the code while submitting the hard copy, just turn it in on Blackboard.

1. Support Vector Machine [13 points]

Let $\phi : \mathbb{R}^d \rightarrow R^D$ be a feature map that transform each input example to a feature vector in $\mathbb{R}^D$, then the primal optimization of SVM is given by

$$\begin{aligned} \underset{\mathbf{w},\xi_i}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i & \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(x_i)) \geq 1 - \xi_i & \forall i = 1, \cdots, n \\ & \xi_i \geq 0 & \forall i = 1, \cdot, n \end{aligned}$$

which is equivalent to the following dual optimization

$$\begin{aligned} \underset{\alpha_i}{\text{minimize}} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) & \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C & \forall i = 1, \cdots, n \\ & \sum_{i=1}^{n}\alpha_i y_i = 0 & \forall i = 1, \cdot, n \end{aligned}$$

where $\xi_1, ..., \xi_n$ are called slack variables. The optimal vector $\mathbf{w}$ can be represented in terms of $\alpha_i, i = 1, \cdots, n$ as $\mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \phi(\mathbf{x}_i)$.

a. Intuitively, where did a data point lie relatively to the margin in the following case (1) $\xi_i = 0$, (2) $0 < \xi_i \leq 1$, and (3) $\xi_i > 1$. Besides, in which case the data

point has been correctly classified? [3 points]
b. Suppose the optimal $\xi_1, ..., \xi_n$ have been computed. Use the ξ_i to obtain an upper bound on the number of misclassified instances. [2 points]
c. In the primal optimization of SVM, what's the role of the coefficient C? Briefly explain your answer by considering two extreme cases, i.e., $C \to 0$ and $C \to \infty$. [4 points]
d. Explain how to use the kernel trick to avoid the explicit computation of the feature vector $\phi(\mathbf{x}_i)$? Also, given a new instance **x**, , how to make prediction on the instance without explicitly computing the feature vector $\phi(\mathbf{x})$? [4 points]

2.Manual calculation of Multinomial Naive Bayes [12 points]
Consider the following training dataset for predicting the class of a document.

Table 1

Doc	Words	Class
1	Russia Moscow	r
2	Washington U.S. Moscow	u
3	Russia St Petersburg	r
4	St Paul U.S.	u
5	U.S. Russia Syria	u

Using a multinomial Naive Bayes classifier, with Laplace (add-1) smoothing, what would be the predicted classes for the following entries? Show your work.

Table 2

Doc	Words	Class
1	Moscow Moscow St Russia	?
2	U.S. St Petersburg	?

3.Perceptron and Kernels [15 points]
(a) Implement the perceptron learning algorithm, as described in class, and run it on the dataset provided (percep1). Keep track of how many iterations you perform until convergence, as well as how many total updates (corresponding to mistakes) that occur through each iteration. After convergence, your code should output the raw weights, as well as the normalized weights corresponding to the linear classifier

$$w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 = 1$$

(You will create the normalized weights by dividing your perceptron weights w_1, w_2, w_3, and w_4 by $-w_0$, the weight corresponding to the special "offset" feature) Report the results of your perceptron learning algorithm as described above. [7 points]

(b) In class, you have seen the primal formulation for perceptron, but there is also the dual form (`http://www.ccs.neu.edu/home/vip/teach/MLcourse/6_SVM_kernels/lecture_notes/kernels/KernelTrick.pdf`). Implement the dual version of perceptron, keeping in mind that you have to make it work with kernels. Apply it to the same dataset as in item a), using a linear kernel, in order to make sure your dual is equivalent to the primal (same solution). Then apply it to a linearly non-separable dataset (percep2) and check the results. Choose an appropriate kernel and report the results. [8 points]

4. SVM [30 points]

In the next two problems, you will perform a digit recognition task using different algorithms. Given an image of a handwritten digit, you want to predict what number it represents. The problem has been simplified to a binary classification between two numbers that sometimes are hard to distinguish (3 vs 5).
The digit images have been taken from a Kaggle competition (`http://www.kaggle.com/c/digit-recognizer/data`) and are originally from the MNIST database of handwritten digits. They are in a format that is easy to read: each row represents an image; the first column is the label and the remaining columns give the grayscale values for each pixel. The version that you will be working on has been filtered to just those datapoints whose labels are 3 or 5 and it has been divided in training and test sets.

Implement SVM with the SMO algorithm and train it on the modified digit dataset. For your implementation, you only have to use the linear kernel. Also run SVM from a package, try different kernels and compare the results. You can implement the simplified SMO, as described in `http://cs229.stanford.edu/materials/smo.pdf`

5. Multi-Layer Perceptron [30 points]

Implement a multilayer perceptron, with one hidden layer, using the backpropagation algorithm and the sigmoid activation function. Run it on the modified digit dataset. Try different number of nodes in the hidden layer, like N=10, 20, 30, 50, 100, and see if there is any difference in the results.