Machine Learning (Spring 2020) **(Due: 16.04.2020 23:55)**

Assignment 4

Instructor: Dr. Sergey Kosov *TA:* Yuhou Zhou

Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- The homework assignments are for practice purpose. The grade from your homework will not affect your final grade of the course.
- Please submit your answer sheet, either by a scanned copy or a typeset PDF file, to Moodle before the deadline.
- No late submission is accepted.
- You can do this assignment in groups of 2. Please submit no more than one submission per group.

Problem 1: Support Vector Machine (5 points)

(a) Show that, irrespective of the dimensionality of the data space, a data set consisting of just two data points, one from each class, is sufficient to determine the location of the maximum-margin hyperplane.

Suppose we have $\mathbf{x}_1$ belongs to class one and we denote its target value $t_1 = 1$, and similarly $\mathbf{x}_2$ belongs to class two and we denote its target value $t_2 = -1$. Since we only have two points, they must have $t_i \cdot y(\mathbf{x}_i) = 1$. Therefore, we have an equality constrained optimization problem:

$$\text{minimize } \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{s.t. } \begin{cases} \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_1) + b = 1 \\ \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_2) + b = -1 \end{cases}$$

The cost function is convex and constrains are affine, so this is an convex optimization problem. It can be solved using Lagrange multiplier.

Note: Read PRML Section 7.1 until the formula (7.4).

Problem 2: Random Forest (5 points)

(a) Show that as the number of the bootstrap samples B gets large, the out-of-bag error for a random forest approaches its N-fold cross validation error estimate, and that in the limit, the identity is exact.

The out-of-bag error is given by

$$\mathcal{E}_{OOB} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Z_i}\sum_{\substack{j \\ (x_i,y_i)\notin D_j}} \ell(h_j(x_i), y_i) \tag{0.1}$$

where D_j is the bagging sample which does not include (x_i, y_i). h_j is the classifier trained on D_j. ℓ is the cost function. Z_i is the number of classifiers not trained on (x_i, y_i).
The LOOCV error for Random Forest is given by

$$\mathcal{E}_{LOOCV} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{B}\sum_{j=1}^{B}\ell(h_{ij}(x_i), y_i) \tag{0.2}$$

where index i of h_{ij} means (x_i, y_i) is the validation data point for the random forest, and index j means the jth tree of the forest.

Assume we have infinite Bagging samples $B = \infty$. The probability that a point is not picked in a bagging sample is $(1 - \frac{1}{N})^N \approx \frac{1}{e}$. $Z_i \approx \frac{B}{e}$.

$$\lim_{B\to\infty} Z_i = \lim_{B\to\infty} \frac{B}{e} = \infty \tag{0.3}$$

so the equation 0.1 and 0.2 are the same.

Problem 3: Artificial Neural Network (5 points)

(a) Consider a convolutional network, in which multiple weights are constrained to have the same value. Discuss how the standard backpropagation algorithm must be modified in order to ensure that such constraints are satisfied when evaluating the derivatives of an error function with respect to the adjustable parameters in the network.

The modifications only affect derivatives with respect to weights in the convolutional layer. The units within a feature map (indexed m) have different inputs, but all share a common weight vector, $\mathbf{w}^{(m)}$. Thus, errors $\delta^{(m)}$ from all units within a feature map will contribute to the derivatives of the corresponding weight vector. In this situation, $\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}}$ becomes

$$\frac{\partial E_n}{\partial w_i^{(m)}} = \sum_j \frac{\partial E_n}{\partial a_j^{(m)}}\frac{\partial a_j^{(m)}}{\partial w_i^{(m)}} = \sum_j \delta_j^{(m)} z_{ji}^{(m)}$$

Here $a_j^{(m)}$ denotes the activation of the j^{th} unit in the m^{th} feature map, whereas $w_i^{(m)}$ denotes the i^{th} element of the corresponding feature vector and, finally, $z_{ji}^{(m)}$ denotes the i^{th} input for the j^{th} unit in the m^{th} feature map; the latter may be an actual input or the output of a preceding layer. Note that $\delta_j^{(m)} = \partial E_n / \partial a_j^{(m)}$ will typically be computed recursively from the δ s of the units in the following layer, using $\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j}$. If there are layer(s) preceding the convolutional layer, the standard backward propagation equations will apply; the weights in the convolutional layer can be treated as if they were independent parameters, for the purpose of computing the δ s for the preceding layer's units.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning: Solutions to the Exercises: Web-Edition.* 2009.

[3] Zhengqi Gao. *Solution Manual for Pattern Recognition and Machine Learning.* 2017.

[4] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2 edition, 2009.