EE6550 Machine Learning, Spring 2017

Homework Assignment #3

Please submit your solutions of the word problems in class (during the first recess time) on April 17th Monday and your programs, including source codes, report (in pdf format) and user manual (in pdf format), to iLMS by 23:59 on April 19th Wednesday. Late submission will not be accepted unless the instructor gives a pre-approval. You are encouraged to consult or collaborate with other students while solving the problems, but you will have to turn in your own solutions and programs with your own words and work. If you find any resources in the internet to assist you, you should understand but not copy them. Copying will not be tolerated.

Part I: Word problem set.

1. (2%) Let $\mathcal{H} \subseteq \{-1, +1\}^{\mathscr{I}}$ be a family of base classifiers over an input space $\mathscr{I}$. Let d be the VC-dimension of $\mathcal{H}$. Let $\mathcal{H}_T$ be the AdaBoost hypothesis set with T rounds of boosting from $\mathcal{H}$, i.e.,

$$\mathcal{H}_T \triangleq \left\{ \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t \right) \mid \alpha_t \in \mathbb{R}, h_t \in \mathcal{H}, t \in [1, T] \right\}.$$

Please use the notations and results you have learned in the lectures of EE6550 to show that

$$\mathrm{VCdim}(\mathcal{H}_T) \leq 2(d+1)(T+1)\log_2((T+1)e).$$

(Hint: Please refer to Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Science* **55**, 1997, pp. 119–139, Theorem 8.)

Part II:

(5%) Programming problem 1:

Implementation of a kernel-based SVM-learning algorithm $\mathbb{A}$ for binary classification by using the kernel-based sequential minimal optimization (SMO) algorithm. Please refer to the article *A Supplement of Kernel-based SMO Algorithm for HW#3 Programming Problem #1*, which will be referred to as *the article*. (Hint: With minor modification, you may transform your SVM-learning algorithm for binary classification implemented in the HW#2 programming problem to a kernel-based SVM-learning algorithm for binary classification.)

(8%) Programming problem 2:

Implementation of the AdaBoost algorithm for binary classification. You should use decision trees of depth one, also known as stumps or boosting stumps as base classifiers for the AdaBoost algorithm.

Input:

Programming problem 1:

1. A data file alphabet_DU_training_header.csv (or alphabet_DU_training.csv) which contains a labeled training sample S. This labeled training sample is used to train the kernel-based SVM-learning algorithm which will return a hypothesis h_S^{SVM} after n-fold cross-validation.
2. A data file alphabet_DU_testing_header.csv (or alphabet_DU_testing.csv) which contains a labeled testing sample $\tilde{S}$. This labeled testing sample is used to evaluate the performance of the returned hypothesis h_S^{SVM} from the kernel-based SVM-learning algorithm based on the labeled training sample S.
3. Choice of the kernel function $K(\mathbf{x}, \mathbf{x}')$. There are two choices:
   (a) The polynomial kernel of degree d: $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^d$. (See pages 13-14 of Lecture 4 on Kernel Methods with $c = 1$.)
   (b) The Gaussian kernel: $K(\mathbf{x}, \mathbf{x}') = \exp\left\{\frac{-\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right\}$.
4. Choice of n-fold cross-validation, where $n = 5$ or $n = 10$. The free model parameter vector $\boldsymbol{\theta}$ is (C, d) if the chosen kernel is the polynomial kernel of degree d and (C, σ) if the chosen kernel is the Gaussian kernel. As discussed in Lecture 1, we use n-fold cross-validation to determine the best value of the free parameter vector $\boldsymbol{\theta}$.
   - Randomly partition a given training sample S of m labeled items into n subsamples or folds.
   - $((\omega_{i1}, c(\omega_{i1})), \ldots, (\omega_{im_i}, c(\omega_{im_i})))$: the ith fold of size m_i, $1 \le i \le n$.
     - Usually $m_i = \frac{m}{n}$ for all i.
   - For any $i \in [1, n]$, the kernel-based SVM-learning algorithm is trained on all but the ith fold to generate a hypothesis h_i, and the performance of h_i is tested on the ith fold.
   - $\hat{R}_{CV}(\boldsymbol{\theta})$: the cross-validation error.

$$\hat{R}_{CV}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i} 1_{h_i(\omega_{ij}) \neq c(\omega_{ij})} = \frac{1}{m}\sum_{i=1}^{n}\sum_{j=1}^{m_i} 1_{h_i(\omega_{ij}) \neq c(\omega_{ij})}.$$

   - Choose a parameter vector $\boldsymbol{\theta}^*$ which minimizes the cross-validation error $\hat{R}_{CV}(\boldsymbol{\theta})$.
   - Train the kernel-based SVM-learning algorithm with the best parameter setting $\boldsymbol{\theta}^*$ over the full training sample S of size m. The resulted hypothesis will be the returned hypothesis h_S^{SVM} from the kernel-based SVM-learning algorithm.

Programming problem 2:

1. A data file alphabet_DU_training_header.csv (or alphabet_DU_training.csv) which contains a labeled training sample S. This labeled training sample is used to train AdaBoost algorithm which will return a hypothesis h_S^{AdaBst} after the n-fold cross-validation.
2. A data file alphabet_DU_testing_header.csv (or alphabet_DU_testing.csv) which contains a labeled testing sample $\tilde{S}$. This labeled testing sample is used to evaluate the performance of the returned hypothesis h_S^{AdaBst}.

3. Choice of n-fold cross-validation, where $n = 5$ or $n = 10$. The free model parameter is the number T of base classifiers needed in the AdaBoost algorithm. As discussed in Lecture 1, we use n-fold cross-validation to determine the best value of the free model parameter T.
   - Randomly partition a given training sample S of m labeled items into n subsamples or folds.
   - $((\omega_{i1}, c(\omega_{i1})), ..., (\omega_{im_i}, c(\omega_{im_i})))$: the ith fold of size m_i, $1 \leq i \leq n$.
     - Usually $m_i = \frac{m}{n}$ for all i.
   - For any $i \in [1, n]$, the AdaBoost algorithm is trained on all but the ith fold to generate a hypothesis h_i, and the performance of h_i is tested on the ith fold.
   - $\hat{R}_{CV}(T)$: the cross-validation error with T base classifiers.

$$\hat{R}_{CV}(T) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m_i}\sum_{j=1}^{m_i} 1_{h_i(\omega_{ij})\neq c(\omega_{ij})} = \frac{1}{m}\sum_{i=1}^{n}\sum_{j=1}^{m_i} 1_{h_i(\omega_{ij})\neq c(\omega_{ij})}.$$

   - Choose the value T^* of T which minimizes the cross-validation error $\hat{R}_{CV}(T)$.
   - Train the AdaBoost algorithm with the best number T^* of base classifiers over the full training sample S of size m. The resulted hypothesis will be the returned hypothesis h_S^{AdaBst} from the AdaBoost algorithm.

Output:

Programming problem 1:

1. The class label mapping. For example, if you map the class label 4 in the alphabet_DU data files to +1 and 21 to −1, you have to output: Class label mapping: 4 → +1 and 21 → −1.
2. The optimal value $\boldsymbol{\theta}^*$ of the free parameter vector $\boldsymbol{\theta}$ with $\boldsymbol{\theta} = (C, d)$ for the polynomial kernel and $\boldsymbol{\theta} = (C, \sigma)$ for the Gaussian kernel (with the precision of the optimal value C^* up to the 2nd decimal point) for both $n = 5$ and $n = 10$ by the n-fold cross-validation.
3. The hypothesis h_S^{SVM} returned by the kernel-based SVM-learning algorithm for both the polynomial kernel and the Gaussian kernel and for both $n = 5$ and $n = 10$ from the n-fold cross-validation,

$$h_S^{SVM}(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{m} \lambda_i^{SVM} c(\mathbf{x}_i) K(\mathbf{x}_i, \mathbf{x}) + b^{SVM}),$$

where λ_i^{SVM} are the values of Lagrangian multipliers λ_i returned by the kernel-based SMO algorithm and the offset b^{SVM} is the sample mean of the offsets

$$b_j{}^{SVM} = c(\mathbf{x}_j) - \sum_{i=1}^{m} \lambda_i^{SVM} c(\mathbf{x}_i) K(\mathbf{x}_i, \mathbf{x}_j)$$

for all support vector $\mathbf{x}_j$ with $0 < \lambda_j^{SVM} < C$, i.e.,

$$\bar{b}^{SVM} = \frac{\sum_{j=1, 0<\lambda_j^{SVM}<C}^{m} b_j^{SVM}}{\sum_{j=1, 0<\lambda_j^{SVM}<C}^{m} 1}.$$

(a) To represent the hypothesis h_S^{SVM} returned from the kernel-based SVM-learning algorithm, please output a .csv file with the file name SVM_hypothesis_header.csv by appending two additional columns with headers *lambda* and *offset* respectively to the training data file alphabet_DU_training_header.csv, which already has one class column and 16 attribute columns, as follows: For the ith item $\mathbf{x}_i$, append λ_i^{SVM} to the *lambda* column and, if $0 < \lambda_i^{SVM} < C$, append b_i^{SVM} to the *offset* column.

(b) Please output the sample mean b^{SVM} and the sample standard deviation $\sigma_{b^{SVM}}$ of the offsets b_j^{SVM},

$$\begin{aligned} b^{SVM} &= \frac{1}{m_b} \sum_{j=1, 0<\lambda_j^{SVM}<C}^{m} b_j^{SVM}, \\ \sigma_{b^{SVM}} &= \sqrt{\frac{1}{m_b - 1} \sum_{j=1, 0<\lambda_j^{SVM}<C}^{m} (b_j^{SVM} - b^{SVM})^2} \end{aligned}$$

where $m_b = \sum_{j=1, 0<\lambda_j^{SVM}<C}^{m} 1$ is the number of items $\mathbf{x}_j$ such that $0 < \lambda_j^{SVM} < C$. (Note: For a good learning result, $\sigma_{b^{SVM}}$ should be small.)

4. Performance of the returned hypothesis h_S^{SVM} on the labeled testing sample $\tilde{S}$ for both the polynomial kernel and the Gaussian kernel and for both $n = 5$ and $n = 10$ in the n-fold cross-validation.

Programming problem 2:

1. The class label mapping. For example, if you map the class label 4 in the alphabet_DU data files to $+1$ and 21 to -1, you have to output: Class label mapping: $4 \rightarrow +1$ and $21 \rightarrow -1$.
2. A table of the cross-validation error $\hat{R}_{CV}(T)$ as a function of the number T of base classifiers and the optimal value of the number T of base classifiers for both $n = 5$ and $n = 10$ by the n-fold cross-validation.
3. The hypothesis h_S^{AdaBst} returned by the AdaBoost algorithm for both $n = 5$ and $n = 10$ from the n-fold cross-validation with

$$h_S^{AdaBst} = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_{S,t}\right),$$

where $h_{S,t}$ is the base classifier learned in the t-th iteration and α_t is the boosting parameter used in the t-th iteration. Since each base classifier $h_{S,t}$ is a univariate stump (a univariate binary tree of depth one), it can be specified by a selected attribute with index i_t, a threshold γ_{i_t}, and an inequality direction, i.e.,

$$h_{S,t}(\mathbf{x}) = \begin{cases} +1, & \text{if } x_{i_t} \geq \gamma_{i_t}, \\ -1, & \text{if } x_{i_t} < \gamma_{i_t}, \end{cases} \quad \text{or} \quad h_{S,t}(\mathbf{x}) = \begin{cases} -1, & \text{if } x_{i_t} \geq \gamma_{i_t}, \\ +1, & \text{if } x_{i_t} < \gamma_{i_t}. \end{cases}$$

To represent the hypothesis h_S^{AdaBst} returned from the Adaboost algorithm, please output a .csv file with the file name AdaBoost_hypothesis_header.csv.

This output file should have five columns with headers *iteration index*, *attribute index*, *threshold*, *direction* and *boosting parameter* respectively. Except the leader line, a line in the output file AdaBoost_hypothesis_header.csv corresponds to the t-th iteratioin of the Adaboost algorithm with t as the iteration index, i_t, λ_{i_t}, and $+1$ or -1 ($+1$ if $x_{i_t} \geq \gamma_{i_t}$ or -1 if $x_{i_t} \geq \gamma_{i_t}$) as the attribute index, the threshold, and the inequality direction of the base classifier $h_{S,t}$, and α_t as the boost parameter.

4. Performance evaluation of the returned hypothesis h_S^{AdaBst} on the labeled testing sample $\tilde{S}$ for both $n = 5$ and $n = 10$ in the n-fold cross-validation.

What to submit? You should submit the following items:

1. The source codes of your kernel-based SVM-learning algorithm based on SMO and your AdaBoost algorithm.
   Please indicate which programming language you have used to write the programs in the title of the submission entry with one of the following formats:
   - HW3_yourstudentid_yourname_matlab (The environment you use should be compatible with the version licensed to the NTHU Computer Center.)
   - HW3_yourstudentid_yourname_python36
   - HW3_yourstudentid_yourname_cpp14
   - HW3_yourstudentid_yourname_c11

   This will facilitate the distribution of homeworks to graders for grading. Please have your code compilable/interpretable by a standard compiler/interpreter environment: Matlab (NTHU CC), Python3.x, C++14, and C11.

2. A report consisting of at least:
   Programming problem 1:
   (a) a class label mapping.
   (b) a table of the cross-validation error $\hat{R}_{CV}(\boldsymbol{\theta})$ as a function of the parameter vector $\boldsymbol{\theta}$ for both the polynomial kernel and the Gaussian kernel and for both $n = 5$ and $n = 10$ by the n-fold cross-validation and a discuss of how you determine the optimal value $\boldsymbol{\theta}^*$ from such a table (with the precision of the optimal parameter C^* up to the 2nd decimal point);
   (c) an output file which represents the hypothesis h_S^{SVM} returned by the kernel-based SVM-learning algorithm as well as the sample mean b^{SVM} and the sample standard deviation $\sigma_{b^{SVM}}$ of the offsets with the optimal parameter vector $\boldsymbol{\theta}^*$ for both the polynomial kernel and the Gaussian kernel and for both $n = 5$ and $n = 10$ from the n-fold cross-validation. In total, there are four such files;
   (d) the performance of the returned hypothesis h_S^{SVM} on the labeled testing sample $\tilde{S}$ for both the polynomial kernel and the Gaussian kernel and for both $n = 5$ and $n = 10$ in the n-fold cross-validation.

   Programming problem 2:
   (a) a class label mapping.
   (b) a table of the cross-validation error $\hat{R}_{CV}(T)$ as a function of the number T of base classifiers for both $n = 5$ and $n = 10$ by the n-fold cross-validation and a discuss of how you determine the optimal value T^* from such a table;

(c) an output file which represents the hypothesis h_S^{AdaBst} returned by the AdaBoost algorithm with the best number T^* of base classifiers for both $n = 5$ and $n = 10$ from the n-fold cross-validation;

(d) the performance of the returned hypothesis h_S^{AdaBst} on the labeled testing sample $\tilde{S}$ for both $n = 5$ and $n = 10$ in the n-fold cross-validation.

Please submit your report in pdf format (please do not submit word files).

3. A user manual which should include instructions of

   (a) how to compile the source code with a standard compiler/interpreter;

   (b) how to run, i.e., execute the kernel-based SVM-learning algorithm and the AdaBoost algorithm, including the required formats of input parameters or data files;

   (c) how to use an output file which represents a returned hypotheses to do the testing.

   (d) what results are reported.

   These instructions should support the test scenarios in the grading session. Please submit your manual in pdf format (please do not submit word files).

Test scenarios in the grading session: the grader will test your algorithm with your source code by the following procedure:
Programming problem 1:

1. inputting a training data file, which contains a labeled training sample S. This labeled training sample is used to train the kernel-based SVM-learning algorithm which will return a hypothesis h_S^{SVM} after the n-fold cross-validation.
2. inputting a testing data file, which contains a labeled testing sample $\tilde{S}$. This labeled testing sample is used to evaluate the performance of the returned hypothesis h_S^{SVM} from the kernel-based SVM-learning algorithm based on the labeled training sample S.
3. choosing a PDS kernel. The free parameter vector $\boldsymbol{\theta}$ is (C, d) for the polynomial kernel and (C, σ) for the Gaussian kernel.
4. inputting a positive integer n to perform n-fold cross-validation to determine the optimal parameter vector $\boldsymbol{\theta}^*$ of the free parameter $\boldsymbol{\theta}$ to minimize the cross-validation error $\hat{R}_{CV}(\boldsymbol{\theta})$.
5. recording the class label mapping.
6. checking the output table of the cross-validation error $\hat{R}_{CV}(\boldsymbol{\theta})$ as a function of the parameter vector $\boldsymbol{\theta}$ and the obtained optimal value $\boldsymbol{\theta}^*$ with a prepared reference table and optimal parameter vector value (with the precision of the optimal parameter C^* up to the 2nd decimal point).
7. inputting the obtained optimal value $\boldsymbol{\theta}^*$ to a prepared reference kernel-based SVM-learning program to return a learned hypothesis h_S^{SVM} represented by an output file grading_SVM_hypothesis_header.csv as well as the sample mean b^{SVM} and the sample standard deviation $\sigma_{b^{SVM}}$ of the offset.
8. checking the output file SVM_hypothesis_header.csv which represents the hypothesis h_S^{SVM} returned by student's kernel-based SVM-learning algorithm as well as the sample mean b^{SVM} and the sample standard deviation $\sigma_{b^{SVM}}$ of

the offset with the file grading_SVM_hypothesis_header.csv generated from the prepared reference program as well as the generated sample mean b^{SVM} and the sample standard deviation $\sigma_{b^{SVM}}$ of the offset.

9. checking the performance of the returned hypothesis h_S^{SVM} on the labeled testing sample $\tilde{S}$ by inputting the student's output file SVM_hypothesis_header.csv to a prepared program.

Programming problem 2:

1. inputting a training data file, which contains a labeled training sample S. This labeled training sample is used to train the AdaBoost algorithm which will return a hypothesis h_S^{AdaBst} after the n-fold cross-validation.
2. inputting a testing data file, which contains a labeled testing sample $\tilde{S}$. This labeled testing sample is used to evaluate the performance of the returned hypothesis h_S^{AdaBst} from the AdaBoost algorithm based on the labeled training sample S.
3. inputting a positive integer n to perform n-fold cross-validation to determine the optimal value T^* of the number T of base classifiers to minimize the cross-validation error $\hat{R}_{CV}(T)$.
4. recording the class label mapping.
5. checking the output table of the cross-validation error $\hat{R}_{CV}(T)$ as a function of the number T of base classifiers and the obtained optimal value T^* of the free parameter T with a prepared reference table and optimal parameter value.
6. inputting the obtained optimal value T^* of the free parameter T to a prepared reference AdaBoost program to return a learned hypothesis h_S^{AdaBst} represented by an output file grading_Adaboost_hypothesis_header.csv.
7. checking the output file Adaboost_hypothesis_header.csv which represents the hypothesis h_S^{AdaBst} returned by student's AdaBoost algorithm with the file grading_Adaboost_hypothesis_header.csv generated from the prepared reference program.
8. checking the performance of the returned hypothesis h_S^{AdaBst} on the labeled testing sample $\tilde{S}$ by inputting the student's output file Adaboost_hypothesis_header.csv to a prepared program.