CptS 570 Machine Learning, Fall 2018
Homework #1
Due Date: Thu, Sept 27 (9:10am)

NOTE 1: Please use a word processing software (e.g., Microsoft word or Latex) to write your answers and submit a printed copy to me at the begining of class on Feb 27. The rationale is that it is sometimes hard to read and understand the hand-written answers.

NOTE 2: Please ensure that all the graphs are appropriately labeled (x-axis, y-axis, and each curve). The caption or heading of each graph should be informative and self-contained.

1. **(5 points)** Answer the following questions with a yes or no along with proper justification.
   a. Is the decision boundary of voted perceptron linear?
   b. Is the decision boundary of averaged perceptron linear?

2. **(5 points)** In the class, we saw the Passive-Aggressive (PA) update that tries to achieve a margin equal to *one* after each update. Derive the PA weight update for achieving margin M.

3. **(10 points)** Consider the following setting. You are provided with n training examples: $(x_1, y_1, h_1), (x_2, y_2, h_2), \cdots, (x_n, y_n, h_n)$, where x_i is the input example, y_i is the class label (+1 or -1), and $h_i > 0$ is the importance weight of the example. The teacher gave you some additional information by specifying the importance of each training example.
   a. How will you modify the perceptron algorithm to be able to leverage this extra information? Please justify your answer.
   b. How can you solve this learning problem using the standard perceptron algorithm? Please justify your answer. I'm looking for a reduction based solution.

4. **(10 points)** Consider the following setting. You are provided with n training examples: $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$, where x_i is the input example, and y_i is the class label (+1 or -1). However, the training data is highly imbalanced (say 90% of the examples are negative and 10% of the examples are positive) and we care more about the accuracy of positive examples.
   a. How will you modify the perceptron algorithm to solve this learning problem? Please justify your answer.
   b. How can you solve this learning problem using the standard perceptron algorithm? Please justify your answer. I'm looking for a reduction based solution.

5. (**10 points**) Suppose we have n_+ positive training examples and n_- negative training examples. Let C_+ be the center of the positive examples and C_- be the center of the negative examples, i.e., $C_+ = \frac{1}{n_+} \sum_{i:\ y_i=+1} x_i$ and $C_- = \frac{1}{n_-} \sum_{i:\ y_i=-1} x_i$. Consider a simple classifier called CLOSE that classifies a test example x by assigning it to the class whose center is closest.

   - Show that the decision boundary of the CLOSE classifier is a linear hyperplane of the form $sign(w \cdot x + b)$. Compute the values of w and b in terms of C_+ and C_-.
   - Recall that the weight vector can be written as a linear combination of all the training examples: $w = \sum_{i=1}^{n_+ + n_-} \alpha_i \cdot y_i \cdot x_i$. Compute the dual weights (α's). How many of the training examples are support vectors?

6. (**5 points**) Suppose we use the following radial basis function (RBF) kernel: $K(x_i, x_j) = exp(-\frac{1}{2}\|x_i - x_j\|^2)$, which has some implicit unknown mapping $\phi(x)$.

   - Prove that the mapping $\phi(x)$ corresponding to RBF kernel has infinite dimensions.
   - Prove that for any two input examples x_i and x_j, the squared Euclidean distance of their corresponding points in the higher-dimensional space defined by ϕ is less than 2, i.e., $\|\phi(x_i) - \phi(x_j)\|^2 \leq 2$.

7. (**5 points**) The decision boundary of a SVM with a kernel function (via implicit feature mapping $\phi(.)$) is defined as follows:
   $w \cdot \phi(x) + b = \sum_{i \in SV} y_i \alpha_i K(x_i, x) + b = f(x; \alpha, b)$
   , where w and b are parameters of the decision boundary in the feature space *phi* defined by the kernel function K, SV is the set of support vectors, and α_i is the dual weight of the i^{th} support vector.

   Let us assume that we use the radial basis function (RBF) kernel $K(x_i, x_j) = exp(-\frac{1}{2}\|x_i - x_j\|^2)$; also assume that the training examples are linearly separable in the feature space ϕ and SVM finds a decision boundary that perfectly separates the training examples.

   If we choose a testing example x_{far} that is far away from any training instance x_i (distance here is measured in the original feature space $\Re^d$). Prove that $f(x_{far}; \alpha, b) \approx b$.

8. (**5 points**) The function $K(x_i, x_j) = -\langle x_i, x_j \rangle$ is a valid kernel. Prove or Disprove it.

9. (**5 points**) You are provided with n training examples: $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n,)$, where x_i is the input example, y_i is the class label (+1 or -1). The teacher gave you some additional information by specifying the costs for different mistakes C_+ and C_-, where C_+ and C_- stand for the cost of misclassifying a positive and negative example respectively.
   a. How will you modify the Soft-margin SVM formulation to be able to leverage this extra information? Please justify your answer.

10. **(5 points)** Consider the following setting. You are provided with n training examples: $(x_1, y_1, h_1), (x_2, y_2, h_2), \cdots, (x_n, y_n, h_n)$, where x_i is the input example, y_i is the class label (+1 or -1), and $h_i > 0$ is the importance weight of the example. The teacher gave you some additional information by specifying the importance of each training example.
    a. How will you modify the Soft-margin SVM formulation to be able to leverage this extra information? Please justify your answer.
    b. How can you solve this learning problem using the standard SVM training algorithm? Please justify your answer.

11. **(15 points)** You are provided with a set of n training examples: $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n,)$, where x_i is the input example, y_i is the class label (+1 or -1). Suppose n is very large (say in the order of millions). In this case, standard SVM training algorithms will not scale due to large training set.
    Tom wants to devise a solution based on "Coarse-to-Fine" framework of problem solving. The basic idea is to cluster the training data; train a SVM classifier based on the clusters (coarse problem); refine the clusters as needed (fine problem); perform training on the finer problem; and repeat until convergence. Suppose we start with k_+ positive clusters and k_- negative clusters to begin with (a cluster is defined as a set of examples). Please specify the *mathematical formulation* (define all the variables used in your formulation) and concrete algorithm for each of the following steps to instantiate this idea:
    a) How to define the SVM training formulation for a given level of coarseness: a set of k_+ positive clusters and a set of k_- negative clusters?

b) How to refine the clusters based on the resulting SVM classifier?
c) What is the stopping criteria?
Optional question: For what kind of problems will this solution fail?

12. **(20 points)** You are provided with a set of n training examples: $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n,)$, where x_i is the input example, y_i is the class label (+1 or -1). Suppose n is very large (say in the order of millions). In this case, online kernelized Perceptron algorithms will not scale if the number of allowed support vectors are *unbounded*.
a) Suppose you have trained using kernelized Perceptron algorithm (without any bounds on support vectors) and got a set of support vectors SV. Tom wants to use this classifier for real-time prediction and cannot afford more than B kernel evaluations for each classification decision. Please give an algorithm to select B support vectors from SV. You need to motivate your design choices in order to convince Tom to use your solution.
b) Tom wants to train using kernelized Perceptron algorithm, but wants to use at most B support vectors during the training process. Please modify the standard kernelized Perceptron training algorithm (from class slides) for this new setting. You need to motivate your design choices in order to convince Tom to use your solution.

13. **(50 points)** Programming and empirical analysis question.
Implement a binary classifier with both perceptron and passive-aggressive (PA) weight update as shown below.

Algorithm 1 Online Binary-Classifier Learning Algorithm

Input: $\mathcal{D}$ = Training examples, T = maximum number of training iterations
Output: w, the final weight vector

Initialize the weights $w = 0$
for each training iteration $itr \in \{1, 2, \cdots, T\}$ **do**
  for each training example $(x_t, y_t) \in \mathcal{D}$ **do**
    $\hat{y}_t = sign(w \cdot x_t)$ *// predict using the current weights*
    if mistake **then**
      $w = w + \tau \cdot y_t \cdot x_t$ *// update the weights*
    end if
  end for
end for
return final weight vector w

For standard perceptron, you will use $\tau = 1$, and for Passive-Aggressive (PA) algorithm, you will compute the learning rate τ as follows.

$$\tau = \frac{1 - y_t \cdot (w \cdot x_t)}{\|x_t\|^2} \tag{1}$$

You are provided with the income data. See `https://archive.ics.uci.edu/ml/datasets/census+income` for a description of different features. You are provided with a training set, development (held out) set, and testing set. The format of the file is as follows. Each line is one classification example: sequence of feature values with the class label at end ($\leq$50K or $>$50K) in comma separated values (CSV) format.

Perceptron, Passive-Aggressive, and Averaged Classifier

- Convert all features into binary format (0 or 1). Describe your conversion. How many features do you have (i.e., the dimensionality)?
- Implement Perceptron and Passive-Aggressive based classifier from these binary features.
- Compute the online learning curve for both Perceptron and PA algorithm by plotting the number of training iterations (1 to 5) on the x-axis and the number of mistakes on the y-axis. Compare the two curves and list your observations.
- Compute the accuracy of both Perceptron and PA algorithm on the training data, development data, and testing data for each training iteration (1 to 5). So you will have three accuracy curves for Perceptron and another three accuracy curves for PA algorithm. Compare the six curves and list your observations.
- Implement the averaged perceptron algorithm in both the naive way (one I wrote on the white board) and the smart way (Algorithm 7 from Hal's book chapter). Measure the training time to perform 5 iterations for both implementations. Compute the accuracies on training data, development data, and testing data with averaged classifier and compare with those obtained using standard perceptron. List your observations.
- Compute the general learning curve (vary the number of training examples starting from 5000 in the increments of 5000) for 5 training iterations. Plot the number of training examples on x-axis and the accuracy (development and testing) on the y-axis. List your observations from these curves.
- Optional things you can try. (a) Shuffling the training examples during each iteration. What did you observe?; and (b) Variable learning rate with a schedule of your choice. What did you observe?

14. **(50 points) Support Vector Machines (SVM) Classifier and Kernels**

- (**30 points**) Empirical analysis question. You can use a publicly available SVM classifier implementation (e.g., LibSVM or scikit-learn) for SVM related experiments. LibSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm/` and scikit-learn (`http://scikit-learn.org/stable/modules/svm.html`).

  (a) Using a linear kernel, train the SVM on the training data for different values of C parameter: $10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}$. Compute the training accuracy, validation accuracy, and testing accuracy for the SVM obtained with different values of the C parameter. Plot the training accuracy, validation accuracy, and testing accuracy as a function of C (C value on x-axis and Accuracy on y-axis) – one curve each for training, validation, and testing data. Also, plot the number of support vectors (see the training log at the end of SVM training for LibSVM or use the scikit-learn API if applicable) as a function of C. List your observations.

  (b) Select the best value of hyper-parameter C based on the accuracy on validation set and train a linear SVM on the *combined* set of training and validation examples. Compute the testing accuracy and the corresponding confusion matrix: a 2×2 matrix.

  (c) Repeat the experiment (a) with the best C value from (a) with polynomial kernel of degree 2, 3, and 4. Compare the training, validation, testing accuracies, and the number of support vectors for differnt kernels (linear, polynomial kernel of degree 2, polynomial kernel of degree 3, and polynomial kernel of degree 4). List your observations.

- (**20 points**) Programming question. You will implement the standard kernelized Perceptron training algorithm (from class slides) for binary classification.

  (a) Train the binary classifier for 5 iterations with polynomial kernel (pick the best degree out of 2, 3, and 4 from the above experiment). Plot the number of mistakes as a function of training iterations. Compute the training, development, and testing accuracy at the end of 5 iterations.

Instructions for Code Submission and Output Format.

Please follow the below instructions. It will help us in grading your programming part of the homework. We will provide a dropbox folder link for code submission.

- Supported programming languages: Python, Java, C++
- Store all the relevant files in a folder and submit the corresponding zipfile named after your student-id, e.g., 114513209.zip
- This folder should have a script file named

  `run_code.sh`

  Executing this script should do all the necessary steps required for executing the code including compiling, linking, and execution
- Assume relative file paths in your code. Some examples:

  ‘‘./filename.txt’’ or ‘‘../hw1/filename.txt’’
- The output of your program should be dumped in a file named “output.txt”
- Make sure the output.txt file is dumped when you execute the script

  `run_code.sh`
- Zip the entire folder and submit it as

  `<student_id>.zip`

Grading Rubric

Each question in the students work will be assigned a letter grade of either A,B,C,D, or F by the Instructor and TAs. This five-point (discrete) scale is described as follows:

- **A) Exemplary (=100%)**.
  Solution presented solves the problem stated correctly and meets all requirements of the problem.
  Solution is clearly presented.
  Assumptions made are reasonable and are explicitly stated in the solution.
  Solution represents an elegant and effective way to solve the problem and is not overly complicated than is necessary.

- **B) Capable (=75%).**
  Solution is mostly correct, satisfying most of the above criteria under the exemplary category, but contains some minor pitfalls, errors/flaws or limitations.

- **C) Needs Improvement (=50%).**
  Solution demonstrates a viable approach toward solving the problem but contains some major pitfalls, errors/flaws or limitations.

- **D) Unsatisfactory (=25%)**
  Critical elements of the solution are missing or significantly flawed.
  Solution does not demonstrate sufficient understanding of the problem and/or any reasonable directions to solve the problem.

- **F) Not attempted (=0%)**
  No solution provided.

The points on a given homework question will be equal to the percentage assigned (given by the letter grades shown above) multiplied by the maximum number of possible points worth for that question. For example, if a question is worth 6 points and the answer is awarded a B grade, then that implies 4.5 points out of 6.