HW6 Boosting, MDP, RL Due Dec 4th midnight

Implementation (Team of up to 3 persons) 55 pts

For this assignment, you will implement the Adaboost ensemble learning algorithm, using Decision stump as the base classifier. To help you structure your code, here are some basic functions you would need to have:

1. Decision_stump_weighted: Given a training set, and a set of weights, learning a decision stump
2. Weighted_error: given a data set with weights, and a classifier (in this case decision stump), calculate the weighted error.
3. Adaboost: the ensemble building algorithm itself. In each iteration, it will call the base learning algorithm to construct a classifier, and evaluated its weighted error on the training data, and construct the updated weights for the next iteration.
4. Test_ensemble: given a learned ensemble (classifiers with their corresponding weights for voting), assess the accuracy or error rate of the ensemble classifier on the given data set.

Please apply your implementation of the boosted decision stump algorithm on the data set provided, changing the ensemble size from 1 to 250. Note that this dataset contains 9 features, which describes what occupies the 9 board positions of a tic-tac-toe game. The class label is binary (1 or 0), indicating the outcome of the game. The data format is in csv, with class labels stored in the first column, and features stored in the remaining columns.

You should provide a report accompanying your source code submission. In your report, you should present the following results in an easily understandable manner:

1. Please plot the training error and test error of the learned ensemble as a function of the ensemble size, varying from 1 to 250. Comment on the performance of the boosted Decision stumps. Do you see significant performance change as we increase the ensemble size? Can you provide any explanation for this behavior?
2. Sometimes we can gauge the importance of the features for predicting the class label based on how frequently each of the features has been selected by the decision stumps in the ensemble. The more frequent, the more important is the feature considered to be. Focusing on ensemble size 1000, please report the frequency of different features being selected by the 100 stumps in your learned ensemble.
3. In boosting each stump is weighted differently, so when a feature is selected by a stump, we can potentially weight its importance by the alpha that the stump receives. Again focusing on the ensemble of size 100, for each feature, please sum up the alphas for all the stumps that test on this feature. Compare the result that you arrive at using this measure of importance with what was produced in (2). Do they differ in terms of your conclusion? What conclusion can you draw
about the features importance from this exercise? Are your conclusions consistent with your understanding of the tic-tac-toe game?

**Written Part (Solo assignment) (45pts)**

1. Consider an undiscounted MDP having three states (1, 2, 3), with rewards (-1, -2, 0) respectively. State 3 is a terminal state. In state 1 and 2 there are two possible actions: a and b. The transition model is as follows:
   - In state 1, action a moves the agent to state 2 with prob. 0.8 and makes the agent stay put with prob. 0.2
   - In state 2, action a moves the agent to state 1 with prob. 0.8 and makes the agent stay put with prob. 0.2
   - In either state 1 or 2, action b moves the agent to state 3 with probability 0.1 and makes the agent stay put with probability 0.9

Answer the following questions.

   a) (10pts) Apply policy iteration (showing each step in full) to determine the optimal policy and the values of states 1 and 2. Assume that the initial policy has action b in both states.
   
   b) (10pts) What happens to policy iteration if the initial policy has action a in both states? What happens if we include a discounting factor, say $\gamma = 0.9$?
   
   c) (5pts) Does the optimal policy depend on the discount factor for this MDP? Does the optimal policy depend on the discount factor in general?

2. You are asked to design an agent to play Tic-Tac-Toe against an opponent who plays randomly. In particular, assume the opponent chooses with uniform probability any open space, unless there is a forced move (in which case it makes the obvious correct move).
   
   a) (10 pts) Formulate an MDP that represents the decision process of playing Tic-Tac-Toe against an opponent who plays randomly. What are the states, actions, transitions, and rewards in this nondeterministic MDP? (note that the opponent is considered to be part of the world)
   
   b) (5 pts) Suppose we do not know the MDP model and applies Q-learning to try to figure out the optimal policy. We represent the Q function as a table, how many entries would it have and why?
   
   c) (5 pts) We train our Q-learner by playing against a random player. After the training is finished, will the learned policy play well against a player who does not play randomly? Why?