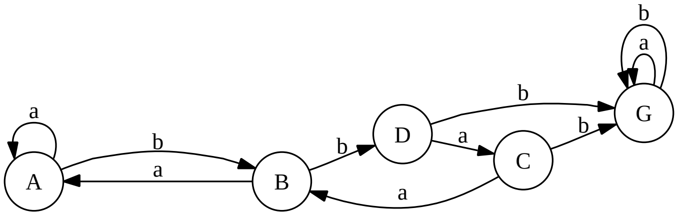
COS 402, Fall 2006
HW #6

Problem 1

Consider the following MDP:



There are five states: A, B, C, D, and G. The reward at every state is -1, except at G where the reward is 0. There are two actions, a and b, and the effect of each action is deterministic as indicated in the figure. For instance, executing a in state B leads to state A. Assume $\gamma = 1$ in this problem.

(a) Show the sequence of utility estimates U_i that would result from executing value iteration on this MDP. Also show the optimal policy that is computed using the final utility estimate.

(b) Show the sequence of policies π_i and utility estimates U^{π_i} that would result from executing policy iteration on this MDP. Assume that you start with a policy that assigns action a to every state. Note that U^{π_i} will be infinite for some states. Also, assume that all ties between the actions a and b in the policy improvement step are always broken in favor of a.

(c) Generalizing this example, suppose we are given a graph with a distinguished node (i.e., state) G, and k edges emanating from every node corresponding to k (deterministic) actions. As in this example, all of the edges emanating from G are self-loops, the node G is assigned reward 0, and all other nodes are assigned reward -1. In terms of properties of the graph, what is the optimal utility function U^*, and what is the optimal policy π^*? If value iteration is applied to this graph (viewed as an MDP), exactly how many iterations will be needed until the algorithm converges? How about for policy iteration?

Problem 2

Let $B(U)$ and $\|\cdot\|_\infty$ be as defined in class. (This is the same as BU and $\|\cdot\|$ defined in Section 17.2 of R&N.) The purpose of this exercise is to prove that B is a *contraction*, i.e., that $\|B(U)-B(U')\|_\infty \leq \gamma\|U-U'\|_\infty$. As discussed in the book and lecture, this is the key step in showing that value iteration converges to the right answer.

We will begin by proving some basic facts. Be sure to give genuine mathematical proofs for each part of this problem. Also, your proofs should use elementary facts — in other words, do not give proofs that rely on mathematical sledge-hammers like the Cauchy-Schwartz inequality.

(a) Let $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ be any sequences of real numbers. Prove that if $u_i \leq v_i$ for all i then

$$\max_i u_i \leq \max_i v_i$$

(b) Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ be any sequences of real numbers. Prove that

$$\left(\max_i x_i\right) - \left(\max_i y_i\right) \leq \max_i (x_i - y_i)$$

and also that

$$\max_i (x_i - y_i) \leq \max_i |x_i - y_i|$$

(Hint: both of these inequalities can be proved using part (a) for an appropriate choice of u_i and v_i.)

Finally, use these facts to prove that

$$\left|\left(\max_i x_i\right) - \left(\max_i y_i\right)\right| \leq \max_i |x_i - y_i|$$

(c) Let $x_1, \ldots, x_n$ be any real numbers, and suppose that $p_1, \ldots, p_n$ are nonnegative real numbers such that $\sum_i p_i = 1$. Use the fact that $|a+b| \leq |a| + |b|$ for any real numbers a and b to prove that

$$\left|\sum_i p_i x_i\right| \leq \max_i |x_i|$$

(d) Now let s be any state, and let $(B(U))(s)$ denote the value of $B(U)$ at state s. By plugging in the definition of B, and using the properties proved above, prove that

$$|(B(U))(s) - (B(U'))(s)| \leq \gamma\|U - U'\|_\infty$$

Conclude that

$$\|B(U) - B(U')\|_\infty \leq \gamma\|U - U'\|_\infty$$

Problem 3

Suppose we start flipping a fair coin. What is the expected number of coin flips until we get three heads in a row? In this problem, we will use a Markov process formulation to solve this problem. The Markov process has six states: Four of the states are HH, HT, TH, and TT, corresponding to the last two times that the coin was tossed. Thus, if we last flipped tails, and the time before that we flipped heads, then we must be in state HT. There is also a "dead" state called D which we reach upon flipping three heads in a row for the first time. Once we reach D, we never exit from it. Finally, the starting state is called S. Thus, we begin in S. After flipping the coin twice, we traverse to one of the four states HH, HT, TH, or TT, corresponding to the outcome of the two flips. Each time the coin is flipped, we traverse to the appropriate state. For instance, if we are in state HT, and the coin comes up tails, then we traverse to TT. Finally, when the coin comes up heads three times in a row, we immediately traverse to the dead state D.

(a) Formulate this problem as a Markov process (i.e., an MDP in which there is no choice of action at each time step) in such a way that the "utility" of every state s is equal to the expected number of coin flips until three heads come up for a random sequence of coin flips beginning in state s. What is the probability of transitioning from every state to every other state? What is the "reward function" at each state?

(b) Write down Bellman-style equations for the utility at each state s in terms of the utilities of the states that can be reached from s in one step.

(c) Solve the system of equations in part (b).

(d) Now answer the original question: What is the expected number of coin flips until we get three heads in a row?

Problem 4

Consider the AdaBoost algorithm in Figure 1 of [1].

(a) Prove that the training error of the final classifier is

$$\frac{1}{M}|\{i : H(x_i) \neq y_i\}| \leq \prod_{t=1}^{T} Z_t.$$

Note that we adopt the notation of the paper: there are M training examples $\{(x_i, y_i) \quad i = 1 \ldots M\}$, T rounds of boosting, and Z_t is defined in Eq. 4 of the paper.

(b) Prove that the following choice of α_t minimizes the training error:

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right),$$

where ϵ_t is the expected proportion of incorrect examples in round t.

Problem 5

Consider the following dataset consisting of five training examples followed by three test examples:

x_1	x_2	x_3	y
training			
−	+	+	−
+	+	+	+
−	+	−	+
−	−	+	−
+	+	−	+
test			
+	−	−	?
−	−	−	?
+	−	+	?

There are three attributes (or features or dimensions), x_1, x_2, and x_3, taking the values + and −. The label (or class) is given in the last column denoted y; it also takes the two values + and −.

Simulate each of the following learning algorithms on this dataset. In each case, show the final hypothesis that is induced, and show how it was computed. Also, say what its prediction would be on the three test examples.

Be sure to see the errata for R&N Chapter 20 below.

(a) *Support vector machines.* For this algorithm, you should interpret both label and attribute values of + and − as the real numbers +1 and −1. Also, you can use the additional information that the first three examples are support vectors, but the others are not, so that α_4 and α_5 are both zero in R&N Eq. (20.17). This means that you can maximize this equation over α_1, α_2, and α_3 using calculus. (Note that if any of these variables turn out to be negative, there's a problem.) When you have found a solution vector $\mathbf{w}$, check it by showing that $y_i(\mathbf{w} \cdot \mathbf{x}_i) \geq 1$, and that equality holds for the support vectors, i.e., the first three examples. (The notation here is as in class and R&N.) You do not need to use a "kernel," just a regular inner product, as in Eqs. (20.17) and (20.18).

(b) *Perceptron.* For this algorithm, use a perceptron with a single output node and the three features at the input level. Attribute values of + and − should be interpreted as the real numbers +1 and −1, while label values of + and

– should be interpreted as 1 and 0. You can disregard the "bias weight" (denoted W_0 in R&N), i.e., assume it is fixed to be zero. Assume that the perceptron is trained for a single epoch that runs through the training data once in the order given. Use a learning rate of $\alpha = 0.1$, and start with all weights equal to zero. For g, use the standard sigmoid function given in Figure 20.16.

Errata for R&N Chapter 20

The paragraph describing SVMs at the very bottom of page 749 continuing at the top of 751 is not quite correct, but some explanation is required to describe what the problem is. In class, we implicitly required the hyperplane sought by the SVM algorithm to pass through the origin. This resulted in a hypothesis of the form

$$\mathrm{sign}(\mathbf{w} \cdot \mathbf{x})$$

In other treatments of SVMs, however, the hyperplane is often *not* required to pass through the origin. Thus, the computed hypothesis has the form

$$\mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

so that the hyperplane is defined both by the vector $\mathbf{w}$ and the scalar b.

The treatment in R&N is not quite correct for either of these cases. For the through-the-origin case, their treatment would be correct if the constraint $\sum_i \alpha_i y_i = 0$ were omitted. With the omission of this constraint, their treatment is the same as was presented in class. For the not-through-the-origin case, the treatment in R&N would be correct if Eq. (20.18) were replaced by

$$h(\mathbf{x}) = \mathrm{sign}\left(\sum_i \alpha_i y_i (\mathbf{x} \cdot \mathbf{x}_i) + b \right)$$

for some b that can be written in terms of the other variables (details omitted). For this class (including Problem 4a above), we will only consider the through-the-origin case.

References

[1] R. Schapire. The boosting approach to machine learning: An overview. *Nonlinear Estimation and Classification*, 2003.