University of Wisconsin – Madison
Computer Sciences Department

CS 760 - Machine Learning

Fall 2011

Exam

11am-12:30pm, Monday, December 5, 2011

CLOSED BOOK
(one sheet of notes, front and back, and a calculator allowed)

Write your answers on these pages and show your work. If you feel that a question is not fully specified, state any assumptions you need to make in order to solve the problem. You may use the backs of these sheets for scratch work. If you use the back for any of your final answers, be sure to clearly mark that on the front side of the sheets.

Neatly write your name on this and all other pages of this exam.

Name ____

Problem	Score	Max Score
1	______	20
2	______	20
3	______	20
4	______	20
5	______	20
TOTAL	______	100

Problem 1 – SVM Learning by the SMO Algorithm (20 points)

You have the small dataset below that involves one feature (A) and the Category (-1 for negative and +1 for positive). Show one update step of the SMO algorithm on this data, assuming the initial alphas and b are 0. Use a linear kernel (dot product). Assume C = 10. Show all your work.

	A	*Category*
Ex1	0	-1
Ex2	1	+1

Problem 2 – Support Vector Machines (20 points)

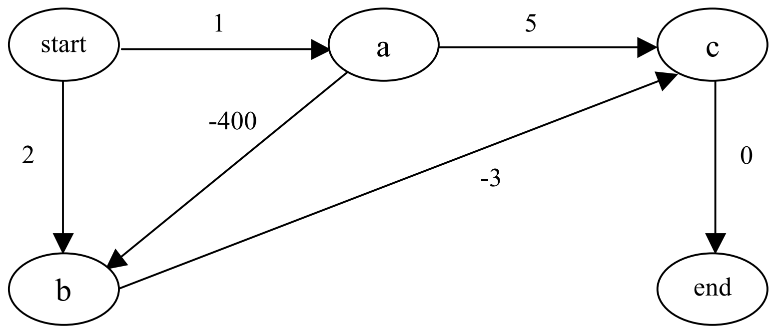
a) What helps support vector machines combat over-fitting, and why does this help?

b) Why can an SVM algorithm learn the exclusive-OR function (XOR, or 2-bit odd parity) of two input binary features if it uses a quadratic kernel but not if it uses a linear kernel? (You may use any representation for binary values that you like in answering this question.)

c) Describe how SVMs are similar to nearest-neighbor classifiers, and how they are different?

Problem 3 – Reinforcement Learning (20 points)

Consider the *deterministic* reinforcement environment drawn below (let $\gamma = 0.1$). The numbers on the arcs indicate the immediate rewards. The probability of an exploration step is 0.01. Assume we learn a *Q-table*. Also assume all the initial values in your *Q* table are 8. (You may assume for your calculations that the 'end' state has only Q-values of 0 for any action).



a) Suppose the learner follows the path *start* → *b* → *c* → *end*. Using <u>one</u>-step, standard *Q* learning, show the calculations that produce the new *Q* table entries and report the final *Q* table on the graph above.

b) After the learning in part (a), the next time the agent begins from the start state, what path will it take to the 'end' state if it performs no exploration steps? Show all the results of one-step, standard Q learning after the agent takes this path.

c) After the learning in part (b), the next time the agent begins from the start state, what path will it take to the 'end' state?

Problem 4 – Inductive Logic Programming (20 points)

Suppose we have an ILP algorithm that performs a **greedy top-down search** of the refinement graph, and we provide it with the positive and negative examples below of the binary predicate grandparent (*gp*). (For example, *gp(a,c)* means *ann* is the grandparent of *carol*.) We also provide it with the background knowledge below of the predicate parent (*p*). (For example, *p(a,b)* means *ann* is the parent of *bob*.) Suppose the **scoring function to be used is P – N**, where P is the number of positive examples covered by a rule, and N is the number of negative examples covered. We do not permit recursive rules (*gp* cannot be used in the body of the rule), and the only arguments to a predicate must be distinct variables; for example, *p(X,Y)* is allowed, but *p(a,X)* and *p(Y,Y)* are not.

Positives	*Negatives*	*Background Knowledge*
gp(a,c)	*gp(a,e)*	*p(a,b)*
gp(b,d)	*gp(b,b)*	*p(b,c)*
gp(c,e)	*gp(e,c)*	*p(c,d)*
		p(d,e)

Show the search that the algorithm will perform, starting with gp(X,Y) at the top of the search. Show the score of each rule that is generated. **Circle** the rule that the algorithm will return at the end.

Name: ______________________

Problem 5–Ensembles (20 points)

How do each of the ensemble approaches below promote diversity in the ensemble?

a) Bagging

b) Random Forests

c) Boosting

d) Error-Correcting Output Codes