CS 188 Spring 2011 Introduction to Artificial Intelligence Practice Final Exam

To earn the extra credit, one of the following has to hold true. Please circle and sign.

A I spent 3 or more hours on the practice final.

B I spent fewer than 3 hours on the practice final, but I believe I have solved all the questions.

Signature: _____

The normal instructions for the exam follow below:

- You have 3 hours.
- The exam is closed book, closed notes except for two double-sided cheat sheets.
- Non-programmable calculators only.
- Mark your answers ON THE EXAM ITSELF. If you are not sure of your answer you may wish to provide a *brief* explanation.

First name	
Last name	
SID	
Login	

For staff use only:

Q1. Search	/14
Q2. Naive Bayes	/14
Q3. Hidden Markov Models	/15
Q4. Machine Learning	/13
Q5. Markov Decision Processes	/10
Q6. Short answer	/19
Total	/85

Q1. [14 pts] Search

For the following questions, please choose the best answer (only one answer per question). **Assume a finite search space.**

(a) [2 pts] Depth-first search can be made to return the same solution as breadth-first search using:

(i) Iterative Deepening

(ii) A closed list/list of nodes that have been expanded

(iii) A heuristic function

(iv) This is not possible

(b) [2 pts] A^* search can be made to perform a breadth-first search by setting (fill in correct values):

1. for all nodes, heuristic =
2. for all nodes, edgecost =

(c) [2 pts] You run A^* search using a heuristic function which you know to be admissible and consistent. Your friend claims he has a search algorithm that is guaranteed to not expand more nodes than your algorithm (and in fact often expands far fewer in practice). He also tells you that his algorithm is guaranteed to find the optimal path.

Could the algorithm your friend claims to have exist? (circle one): yes no
Explain:

(d) [2 pts] Depth first search using a closed list/list of nodes that have been expanded is:

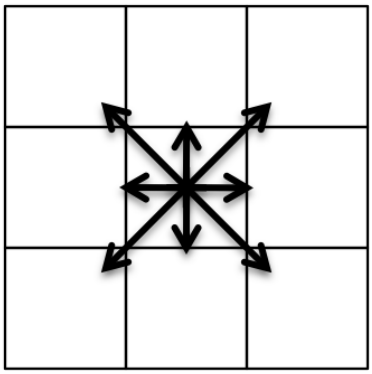
(i) Optimal (will find a shortest path to goal)

(ii) Complete (will find a path to goal if at least one exists)

(iii) Both optimal and complete

(iv) Neither optimal nor complete

Consider a grid, a portion of which is shown below:



You would like to search for paths in this grid. Unlike in Pacman, it is possible to move diagonally as well as horizontally and vertically. The distance between neighboring grid squares (horizontally or vertically) is 1, and the distance between diagonally adjacent grid squares is $\sqrt{2}$.

(e) [2 pts] Is the euclidean distance an admissible heuristic? The euclidean distance between two points (x_1, y_1) and (x_2, y_2) is $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

(f) [2 pts] The Manhattan distance is not an admissible heuristic. Can it be made admissible by adding weights to the x and y terms? The Manhattan distance between two points (x_1, y_1) and (x_2, y_2) is $|x_2 - x_1| + |y_2 - y_1|$. A weighted version with weights α and β would be $\alpha|x_2 - x_1| + \beta|y_2 - y_1|$. Specify the (possibly empty) set of pairs of weights (α, β) such that the weighted Manhattan distance is admissible.

(g) [2 pts] Is the L_∞ distance an admissible heuristic? The L_∞ distance between two points (x_1, y_1) and (x_2, y_2) is $max(|x_2 - x_1|, |y_2 - y_1|)$

Q2. [14 pts] Naive Bayes

Your friend claims that he can write an effective Naive Bayes spam detector with only three features: the hour of the day that the email was received ($H \in \{1, 2, \ldots, 24\}$), whether it contains the word 'viagra' ($W \in \{\text{yes}, \text{no}\}$), and whether the email address of the sender is Known in his address book, Seen before in his inbox, or Unseen before ($E \in \{\text{K}, \text{S}, \text{U}\}$).

(a) [3 pts] Flesh out the following information about this Bayes net:

Graph structure:

Parameters:

Size of the set of parameters:

Suppose now that you labeled three of the emails in your mailbox to test this idea:

spam or ham?	H	W	E
spam	3	yes	S
ham	14	no	K
ham	15	no	K

(b) [2 pts] Use the three instances to estimate the maximum likelihood parameters.

(c) [2 pts] Using the maximum likelihood parameters, find the predicted class of a new datapoint with $H = 3$, $W = \text{no}$, $E = U$.

(d) [4 pts] Now use the three training examples to estimate the parameters using Laplace smoothing and $k = 2$. Do not forget to smooth both the class prior parameters and the feature values parameters.

(e) [3 pts] You observe that you tend to receive spam emails in batches. In particular, if you receive one spam message, the next message is more likely to be a spam message as well. Explain a new graphical model which most naturally captures this phenomena.

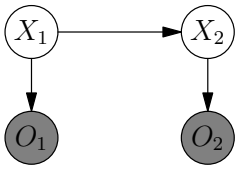
Graph structure:

Parameters:

Size of the set of parameters:

Q3. [15 pts] Hidden Markov Models

Consider the following Hidden Markov Model.



X_1	$\Pr(X_1)$
0	0.3
1	0.7

X_t	X_{t+1}	$\Pr(X_{t+1}\|X_t)$
0	0	0.4
0	1	0.6
1	0	0.8
1	1	0.2

X_t	O_t	$\Pr(O_t\|X_t)$
0	A	0.9
0	B	0.1
1	A	0.5
1	B	0.5

Suppose that $O_1 = A$ and $O_2 = B$ is observed.

(a) [3 pts] Use the Forward algorithm to compute the probability distribution $\Pr(X_2, O_1 = A, O_2 = B)$. Show your work. You do not need to evaluate arithmetic expressions involving only numbers.

(b) [3 pts] Compute the probability $\Pr(X_1 = 1 | O_1 = A, O_2 = B)$. Show your work.

For the next two questions, use the specified sequence of random numbers $\{a_i\}$ generated independently and uniformly at random from $[0, 1)$ to perform sampling. Specifically, to obtain a sample from a distribution over a variable $Y \in \{0, 1\}$ using the random number a_i, pick $Y = 0$ if $a_i < \Pr(Y = 0)$, and pick $Y = 1$ if $a_i \geq \Pr(Y = 0)$. Similarly, to obtain a sample from a distribution over a variable $Z \in \{A, B\}$ using the random number a_i, pick $Z = A$ if $a_i < \Pr(Z = A)$, and pick $Z = B$ if $a_i \geq \Pr(Z = A)$. Use the random numbers $\{a_i\}$ in order starting from a_1, using a new random number each time a sample needs to be obtained.

(c) [3 pts] Use likelihood-weighted sampling to obtain 2 samples from the distribution $\Pr(X_1, X_2|O_1 = A, O_2 = B)$, and then use these samples to estimate $\mathrm{E}[\sqrt{X_1 + 3X_2}|O_1 = A, O_2 = B]$.

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
0.134	0.847	0.764	0.255	0.495	0.449	0.652	0.789	0.094	0.028

(d) [2 pts] [*true* or *false*] In general, particle filtering using a single particle is equivalent to rejection sampling in the case that there is no evidence. Explain your answer.

(e) [2 pts] [*true* or *false*] Performing particle filtering twice, each time with 50 particles, is equivalent to performing particle filtering once with 100 particles. Explain your answer.

(f) [2 pts] [*true* or *false*] Variable elimination is generally more accurate than the Forward algorithm. Explain your answer.

Q4. [13 pts] Machine Learning

(a) [4 pts] Indicate which, if any, of the statements are correct, and explain your answer. Naive Bayes trained using maximum-likelihood parameter estimation

(i) is guaranteed not to perform worse on the test set if more features are added.

(ii) generally performs better on the training set if add-k smoothing is used.

(b) [2 pts] For data points that are *not* normalized (normalized means that for all data vectors x, $||x|| = 1$), we can implement nearest neighbor classification using the distance metric $||x - y||$ defined between two data vectors x and y to determine the nearest neighbor of each test point. Given a kernel function $K(x, y)$, describe how to implement a kernelized version of this algorithm.

Hint: A kernel function $K(x, y)$ is a function that computes the inner product of the vectors x and y in another (typically higher dimensional) space, i.e.: $K(x, y) = \phi(x) \cdot \phi(y)$ The kernel function usually uses an efficient shortcut for the above expression and does not compute $\phi(x)$ and $\phi(y)$ which could be inefficient or even impossible. Our goal is to compute $d(x, y) = ||\phi(x) - \phi(y)||$ without actually using the projection function ϕ but just using the kernel function. Notice that for a pair of vectors a and b, $||a - b||^2 = (a - b) \cdot (a - b)$.

(c) [2 pts] Indicate which, if any, of the statements are correct, and explain your answer. Assuming a linearly separable dataset, Support Vector Machines typically:

(i) when evaluated on the training set, achieve higher accuracy than Perceptrons.

(ii) when evaluated on the test set, achieve higher accuracy than Perceptrons because SVMs can be used with kernels.

(d) [3 pts] Consider the problem of detecting human faces in images of larger scenes that may contain one or more faces, or no faces at all. The specific goal is to identify the *location and size* of each face in an image, rather than merely determining *whether* a face is present in the image. Suppose you are given as training data a set of images in which all of the faces have been labeled (specified as a list of pixel positions for each face). Describe one approach to solving this problem using *binary classification* machine learning. Specify what will serve as training examples for the binary classification algorithm, and how the binary classifier can be used to detect faces in new images. You don't need to specify the specific features or classification algorithm to use.

(e) [2 pts] [*true* or *false*] In the case of a binary class and all binary features, Naive Bayes is a linear classifier. *Briefly* justify your answer.

Q5. [10 pts] Markov Decision Processes

(a) In the standard MDP formulation, we define the utility of a state-action sequence to be the sum of discounted rewards obtained from that sequence, i.e.

$$U_+(s_0, a_0, s_1, \ldots, a_n, s_n) = \sum_{i=0}^{n-1} \gamma^i R(s_i, a_i, s_{i+1}),$$

and the goal is to obtain a policy that maximizes the expected utility of the resultant state sequence. Under this utility function, the optimal policy depends only on the current state and not on any previous states or actions.

Suppose that we redefine the utility of a state-action sequence to be the *maximum* reward obtained from that sequence, i.e.

$$U_{\max}(s_0, a_0, s_1, \ldots, a_n, s_n) = \max_{i=0}^{n-1} R(s_i, a_i, s_{i+1}).$$

We still wish to obtain a policy that maximizes the expected utility of the resultant state sequence.

(i) [3 pts] Show by way of a counter-example that with this modified utility function, the optimal policy does not depend only on the current state.

(ii) [7 pts] Suppose you are given an arbitrary MDP $M = (S, A, T, R)$ with the *modified utility function* $U_{\max}$, where S denotes the state set, A denotes the action set, $T(s, a, s')$ denotes the transition probability function, and $R(s, a, s')$ denotes the reward function. Define a corresponding MDP $M' = (S', A', T', R', \gamma')$ with the property that the optimal policy for M' under the standard utility function U_+ specifies the optimal policy for M under the modified utility function $U_{\max}$.

$S' =$

$A' =$

Transition function:

Reward function:

$\gamma' =$

Q6. [19 pts] Short answer

Each true/false question is worth 1 point. Leaving a question blank is worth 0 points. **Answering incorrectly is worth −1 point.**

(a) If $h_1(s)$ and $h_2(s)$ are two consistent A^* heuristics, then

(i) [*true* or *false*] $h(s) = \frac{1}{2}h_1(s) + \frac{1}{2}h_2(s)$ must also be consistent.

(ii) [*true* or *false*] $h(s) = \max(h_1(s), h_2(s))$ must also be consistent.

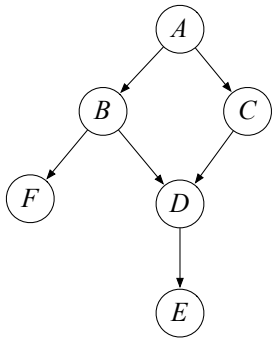
(iii) [*true* or *false*] $h(s) = \min(h_1(s), h_2(s))$ must also be consistent.

(b) Assume we want to solve two search problems, which share the search graph (and associated step costs) and the start state, but the goal state (and heuristic) are different. Let h_1 and h_2 be the heuristics for goals G_1 and G_2 respectively.

(i) [*true* or *false*] If h_1 and h_2 are consistent, we can run A* graph search with a heuristic $h(s) = \min(h_1(s), h_2(s))$ and find solutions to both problems by simply running the search until both G_1 and G_2 have been popped from the queue.

(ii) [*true* or *false*] If h_1 and h_2 are consistent, we can first run A* graph search for G_1. While doing so we can cache all pathcost(s) values popped from the queue and use them to initialize the search queue for G_2. This will result in the search for G_2 finding the optimal solution.

(c) In the Bayes Net to the right, which of the following conditional independence assertions are true?



(i) [*true* or *false*] $A \perp\!\!\!\perp E$

(ii) [*true* or *false*] $B \perp\!\!\!\perp C|A$

(iii) [*true* or *false*] $F \perp\!\!\!\perp C|A$

(iv) [*true* or *false*] $B \perp\!\!\!\perp C|A, E$

(d) In variable elimination,

(i) [*true* or *false*] the ordering of variables in variable elimination affects the maximum factor size generated by at most a factor of two.

(ii) [*true* or *false*] the size of factors generated during variable elimination is upper-bounded by twice the size of the largest conditional probability table in the original Bayes net.

(iii) [*true* or *false*] the size of factors generated during variable elimination is the same if we exactly reverse the elimination ordering.

(e) Consider two particle filtering implementations:

Implementation 1:
Initialize particles by sampling from initial state distribution and assigning uniform weights.

1. Propagate particles, retaining weights
2. Resample according to weights
3. Weight according to observations

Implementation 2:
Initialize particles by sampling from initial state distribution.

1. Propagate unweighted particles
2. Weight according to observations
3. Resample according to weights

(i) [*true* or *false*] Implementation 2 will typically provide a better approximation of the estimated distribution than implementation 1.

(ii) [*true* or *false*] If the transition model is deterministic then both implementations provide equally good estimates of the distribution.

(iii) [*true* or *false*] If the observation model is deterministic then both implementations provide equally good estimates of the distribution.

(f) Particle filtering:

(i) [*true* or *false*] It is possible to use particle filtering when the state space is continuous

(ii) [*true* or *false*] It is possible to use particle filtering when the state space is discrete

(iii) [*true* or *false*] As the number of particles goes to infinity, particle filtering will represent the same probability distribution you would get with exact inference

(iv) [*true* or *false*] Particle filtering can represent flat (i.e. uniform) distribution using fewer particles than it would need for a more concentrated distribution (i.e. Gaussian)