Chun-Wei Chen

1267040

CSE 473 Take-Home Final

1.

(a) A pattern database is essentially a heuristic function for solving problems by searching. The idea behind pattern databases is to store the exact solution costs for every possible subproblem instance into a lookup table. The reason why storing the cost of subproblems is because the cost of subproblems is a lower bound, which is admissible, of the cost of the complete problem, and it often serve as good estimate of the overall solution cost. The database itself is constructed by searching back from the goal and recording the cost of each new pattern encountered. After storing the cost of subproblems, we can compute an admissible heuristic for each complete state encountered during a search simply by looking up the corresponding subproblem configuration. Pattern databases have been widely used in game playing simply because it provides a huge amount of speed up compare to using simple heuristic function, such as Manhattan distance. For instance, solving 15 Puzzle using disjoint pattern database is 2000 times faster than using Manhattan distance.

(b) The idea of using simulated annealing is to search for feasible solutions and converge to an optimal solution. A hill climbing algorithm that never makes “downhill” moves toward states with lower state (or higher cost) is guaranteed to be incomplete since it can get stuck on a lock maximum. Simulated annealing solves the problem of getting stuck on local maxima by allowing worse moves (lesser quality) to be taken with some probability. That is, it allows some uphill steps so that it can escape from local minima. If the move is better than its current position then simulated annealing will always take it. If the move is worse than the current position, the algorithm will accept the move based on some probability. The probability starts high and gets lower after each move. Simulated annealing can be proved that it will find a global optimum with probability approaching 1 if T, variable controlling probability of downward steps, decreases slowly enough. Simulated annealing is widely used for optimizing VLSI layout, airline scheduling, etc.

(c) An expectiminimax tree is a specialized variation of a minimax game tree for use in artificial intelligence systems that play two-player zero-sum games such as backgammon, in which the outcome depends on a combination of the player's skill and chance elements such as dice rolls. In addition to "min" and "max" nodes of the traditional minimax tree, this variant has "chance" ("move by nature") nodes, which take the expected value of a random event occurring. In game theory terms, an expectiminimax tree is the game tree of an extensive-form game of perfect, but incomplete information. Instead of taking the max or min of the utility values of their children, chance nodes take a weighted average, with the weight being the probability that that child is reached. Expectiminimax algorithm should be used for non-deterministic games instead of minimax algorithm since we need to consider the probability of taking the moves other than the best move.

(d) Skolemization is the process of removing existential quantifiers by elimination. The simplest form of Skolemization is for existentially quantified variables which are not inside the scope of a universal quantifier. These can simply be replaced by creating new constants. For instance $\exists x P(x)$ can be changed to $P(c)$, where c is a new constant that does not occur anywhere else in the formula. More generally, Skolemization is performed by replacing every existentially quantified variable, say y, with a function, say f(x), in the scope of the existential quantifier, $\exists y$. Such function used to remove existential quantifiers is called Skolem functions. The general rule is that the argument in whose scope the Skolem function are all the universally quantified variables in whose scope the existential quantifier appears. One of the uses of Skolemization is automated theorem proving program, such as SNARK. The Skolemization is useful to answer a query or prove a theorem because the Skolemized sentence is satisfiable exactly when the original sentence is satisfiable.

(e) Each individual learning technique might yield a distinct hypothesis (or function), and there is no perfect learning hypothesis. The idea of ensemble classification techniques is to select a collection, or ensemble, of hypotheses from the hypothesis space and combine their predictions. One technique of ensemble learning is called bagging. The intuition of bagging technique is that individuals makes mistake but the majority may be less likely to, and individuals often have partial knowledge while a committee can pool expertise to make better decisions. Bagging combines hypotheses via majority voting; therefore, the probability of misclassifying a data will go down by using bagging. Another technique is called boosting, which increase an input's weight when it is misclassified so that the next classifier is more likely to classify it correctly. One of the most importance applications of ensemble classification techniques is face detection. Most modern cameras have face detection so that the picture it takes can focus on people's face instead of focusing on so other places in the picture.

(f) In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the *k* subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data. The cross-validation process is then repeated *k* times (the folds), with each of the *k* subsamples used exactly once as the validation data. The *k* results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. Similar to the idea of ensemble learning, the more time we validate, we are more likely to get the accurate hypothesis. Popular values for k are 5 and 10 – enough give an estimate that is statistically likely to be accurate, at a cost of 5 to 10 times longer computation.

2. (a) $P(Fuel = Yes \mid FM = Empty)$

$= P(FM = Empty \mid Fuel = Yes)\, P(Fuel = Yes) \,/\, P(FM = Empty)$

$= \frac{(1-0.4-0.4)\times 0.6}{[(1-0.4-0.4)\times 0.6 + (1-0.1-0.05)\times 0.4]}$

$= \frac{0.12}{0.12+0.34} \approx 0.26$

(b) $P(FM, St, Fuel, SP)$

$= P(FM \mid St, Fuel, SP)\, P(St, Fuel, SP)$

$= P(FM \mid Fuel)\, P(St, Fuel, SP)$

$= P(FM \mid Fuel)\, P(St \mid Fuel, SP)\, P(Fuel, SP)$

$= P(FM \mid Fuel)\, P(St \mid Fuel, SP)\, P(Fuel)\, P(SP)$

(c) $P(Fuel = No, SP = Yes, FM = Half, St = No)$

$= P(FM = Half \mid Fuel = No)\, P(St = No \mid Fuel = No, SP = Yes)\, P(Fuel = No)\, P(SP = Yes)$

$= 0.1 \times (1-0.01) \times 0.4 \times 0.8$

$= 0.03$

(d) $P(St = Yes \mid FM = Empty) = \frac{P(St = Yes, FM = Empty)}{P(FM = Empty)}$

$P(St = Yes, FM = Empty) = \sum_f P(FM = Empty \mid Fuel = f)\, P(Fuel = f) \sum_{sp} P(SP = sp)\, P(St = Yes \mid Fuel = f, SP = sp)$

$= (1-0.4-0.4)\times 0.6 \times (0.95\times 0.8 + 0.1\times 0.2) + (1-0.1-0.05)\times 0.4 \times (0.01\times 0.8 + 0\times 0.2)$

$= 0.0936 + 0.00272 = 0.09632$

We've got $P(FM = Empty)$ in (a), which is $0.12 + 0.34 = 0.46$

$\therefore P(St = Yes \mid FM = Empty) = \frac{0.09632}{0.46} \approx 0.21$

(e) $P(FM = fm \mid St = No) = \frac{1}{P(St = No)}\, P(FM = fm, St = No) = \frac{1}{P(St = No)} \sum_f P(FM = fm \mid Fuel = f)\, P(Fuel = f) \sum_{SP} P(SP = sp)\, P(St = No \mid Fuel = f, SP = sp)$

First, eliminate SP

Fuel	SP	P(St=No)
Y	Y	0.05
Y	N	0.90
N	Y	0.09
N	N	1

SP	P(SP)
Y	0.8
N	0.2

$\Rightarrow$

Fuel	P(St=No)
Y	$0.8\times 0.05 + 0.2\times 0.90 = 0.22$
N	$0.8\times 0.09 + 0.2\times 1 = 0.272$

Then, eliminate Fuel

Fuel	P(Fuel)
Y	0.6
N	0.4

Fuel	P(St=No)
Y	0.22
N	0.272

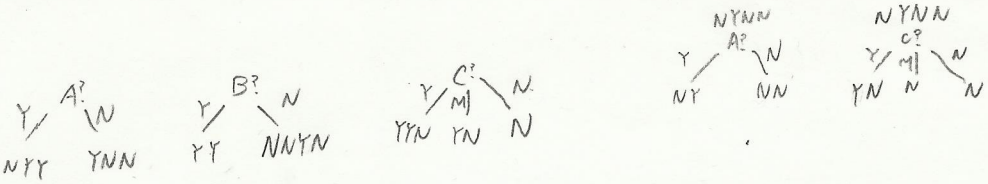
Fuel	P(FM=Full)	P(FM=Half)	P(FM=Empty)
Y	0.40	0.40	0.20
N	0.05	0.10	0.85

$\Rightarrow$

FM	P(FM, St=No)
Full	$0.6\times 0.22\times 0.40 + 0.4\times 0.272\times 0.05 = 0.05824$
Half	$0.6\times 0.22\times 0.40 + 0.4\times 0.272\times 0.10 = 0.06368$
Empty	$0.6\times 0.22\times 0.20 + 0.4\times 0.272\times 0.85 = 0.11888$

Finally, divide by $P(St = No)$ to get $P(FM = Empty \mid St = No)$. $P(St = No) = 0.22\times 0.6 + 0.272\times 0.4 = 0.2408$

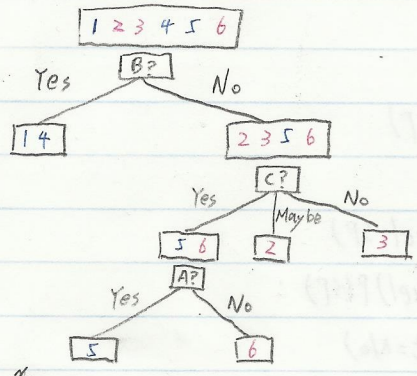
FM	P(FM | St=No)
Full	$0.05824 / 0.2408 \approx 0.24186$
Half	$0.06368 / 0.2408 \approx 0.26445$
Empty	$0.11888 / 0.2408 \approx 0.49369$



3. $IG(A) = 1 - 0.55 = 0.45$, $IG(B) = 1 - 0.16 = 0.84$, $IG(C) = 1 - 0.24 = 0.76$.

∴ Choose B at the root is the best

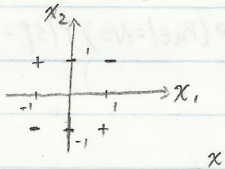
At the second level, $IG(A) = IG(C)$, I'll choose C at the second level.



Number in —: output Yes
Number in —: output No

4. (a)

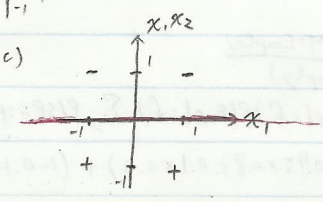
x_1	x_2	XOR
+1	+1	-1
+1	-1	+1
-1	+1	+1
-1	-1	-1



(b)

x_1	x_1x_2	XOR
+1	+1	-1
+1	-1	+1
-1	-1	+1
-1	+1	-1

(c)



(d) I draw the maximum margin seperating line in red on graph for (c)

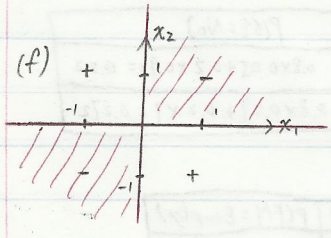
the equation is $x_1x_2 = 0$

(e) maximum margin = distance between either one of the four data point to the line $x_1x_2 = 0$

$= \|(1,1) - (1,0)\|$

$= 1$

(f)



The seperating line in (d) is two intersecting lines in original 2D space.

Two intersecting lines are $x_1 = 0$ and $x_2 = 0$.

The output is -1 when $x_1x_2 > 0$ and +1 when $x_1x_2 < 0$