Name:

Birth Date:

Matriculation Number:

Exam
Artificial Intelligence 1

July 17., 2018

	To be used for grading, do not write here												
prob.	1.1	1.2	2.1	2.2	2.3	3.1	3.2	3.3	4.1	5.1	5.2	Sum	grade
total	4	12	4	3	12	4	10	12	10	10	8	89	
reached													

Exam Grade: Bonus Points: Final Grade:

The “solutions” to the exam/assignment problems in this document are supplied to give students a starting point for answering questions. While we are striving for helpful “solutions”, they can be incomplete and can even contain errors.

If you find “solutions” you do not understand or you find incorrect, discuss this on the course forum and/or with your TA and/notify the instructors.

In any case, grading student’s answers is not a process of simply “comparing with the reference solution”.

In the course Artificial Intelligence I/II we award 5 bonus points for the first student who reports a factual error (please report spelling/formatting errors as well) in an assignment or old exam and 10 bonus points for an alternative solution (formatted in LaTeX) that is usefully different from the existing ones.

1 Prolog

Problem 1.1 (The Zip Function)

The zip function takes two lists with lengths that differ at most by 1, and outputs a list of lists containing one element from the first list and the element with the same index from the other list, possibly followed by a one-element list with the left-over argument. 4 pt 4 min

Create a ProLog predicate with 3 arguments: the first two would be the two lists you want to zip, and the third one would be the result. For instance:

```
?- zip([1,2,3],[4,5,6],L).      ?- zip([1,2],[3,4,5],L).
L = [[1, 4], [2, 5], [3, 6]].   L = [[1, 3], [2, 4], [5]].
```

Feel free to implement any helper functions.

Hint: Remember that you can pattern match a list L as [HEAD|TAIL].

Solution:

```
zip([L],[],[[L]]).
zip([],[L],[[L]]).
zip([A],[B],[[A,B]]).
zip([H1|T1],[H2|T2],L) :- zip(T1,T2,T), append([[H1,H2]],T,L).
```

Problem 1.2 (DFS in Prolog)

We want to implement DFS in ProLog using the following data structures for search trees: 12 pt 12 min

```
subtrees([]).
subtrees([(Cost,T)|Rest]) :- number(Cost),istree(T), subtrees(Rest).
istree(tree(_,Children)) :- subtrees(Children).
```

Write a Prolog predicate dfs such that dfs(G,T,X,Y) on a tree T returns the path to the goal G in X and the cost of the path in Y

Solution:

```
dfs(GoalValue,tree(GoalValue,_),GoalValue,0).
dfs(GoalValue,tree(Value,[(Cost,T)|Rest]),Path,FinalCost) :- T = tree(IV,_), write(IV ),
dfs(GoalValue, T,P,C),string_concat(Value,P,Path),FinalCost is C+Cost; % go down one depth level
dfs(GoalValue,tree(Value,Rest),Path,FinalCost). % next child
```

2 Bayesian Reasoning

Problem 2.1 (Bayesian Rules)

Name four of the basic rules in Bayesian inference and explain each with a short sentence and formula. 4 pt 4 min

Solution:

1. Bayes rule (compute $P(A|B)$ from $P(B|A)$,

2. Normalization (Fixing evidence e, updating the probabilities of all other events using a normalization constant α),
3. Marginalization ($P(A) = \sum_y P(A, y)$),
4. Chain rule ($P(A_1, \ldots, A_n) = P(A_n | A_{n-1}, \ldots, A_1) \cdot P(A_{n-1} | A_{n-2}, \ldots, A_1) \cdot \ldots$)
5. Product rule ($P(A, B) = P(A|B)P(B)$)
6. Conditional Independence (Not really bayesian inference, but rather bayesian networks, but we'll be lenient)

Problem 2.2 (Conditional Independence)

Define *conditional independence.* 3 pt

Solution:Two events A, B are conditionally independent given C, if $P(A \wedge B|C) = P(A|C)P(B|C)$. 3 min

Problem 2.3 (Medical Bayesian Network 2)

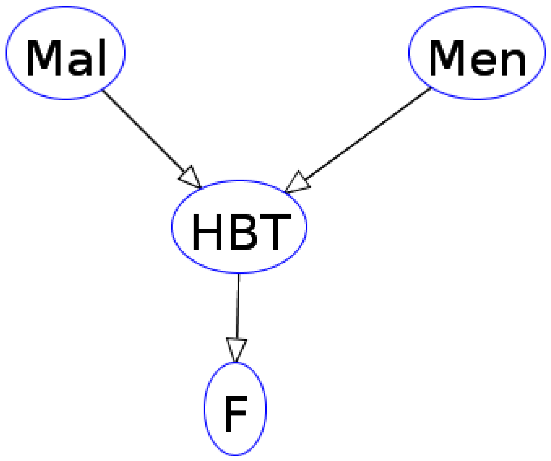
Both Malaria and Meningitis can cause a fever, which can be measured by checking for a high body temperature. Of course you may also have a high body temperature for other reasons. We consider the following random variables for a given patient: 12 pt 12 min

- Mal: The patient has malaria.
- Men: The patient has meningitis.
- HBT: The patient has a high body temperature.
- F: The patient has a fever.

1. Draw the corresponding Bayesian network for the above data using the algorithm presented in the lecture, assuming the variable order Mal, Men, HBT, F. Explain rigorously(!) the exact criterion for whether to insert an arrow between two nodes.
2. Which arrows are causal and which are diagnostic? Which order of variables would be better suited for constructing the network?
3. How do we compute the probability the patient has malaria, given that he has a fever? State the query variables, hidden variables and evidence and write down the equation for the probability we are interested in.

Solution:

1. The following graph but with Edges from Mal and Men to F because it is much more likely that F is the cause of HBT if someone is actually sick.



Let $Parents(X)$ be the minimal set of previous events Y such that $P(X|Parents(X)) = P(X|Y)$. We draw an arrow from all events in $Parents(X)$ to X.

We start with Mal. Continuing with Men, we don't insert an arrow, since Mal and Men are independent $P(Men|Mal) = P(Men)$. Continuing with HBT, we have $Parents(HBT) = \{Mal, Men\}$. Continuing with F, we have $P(F|HBT) = P(F|HBT, Men, Mal)$.

2. The arrows $Mal \to HBT$ and $Men \to HBT$ are causal, the arrow $HBT \to F$ is diagnostic.
3. Query variable: Mal. Evidence: F. Hidden variables: v_{Men}, v_{HBT}. We get:

$$P(Mal|F) = \alpha \sum_{v_{HBT}, v_{Men}} P(Mal) \cdot P(v_{Men}) \cdot P(v_{HBT}|Mal, v_{Men}) \cdot P(F|v_{HBT})$$

---

3 Decision Theory

Problem 3.1 (Expected Utility)
What is the formal(!) definition of *expected utility*? Explain every variable in the defining equation. 4 pt

4 min

Solution:The expected utility EU is defined as

$$EU(a|e) = \sum_{s'} P(R(a) = s'|a, e) \cdot U(s')$$

where

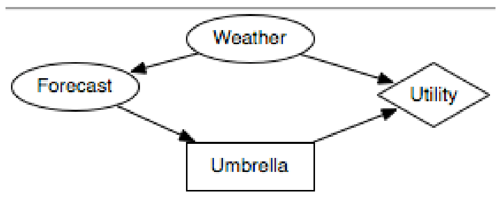
1. a is the action for which we want to find out the expected utility, given the evidene e.
2. $U(s')$ is the utility of a state s'.
3. $R(a)$ is the result of the action a.

Problem 3.2 (Decision Network)

You try to decide on whether to take an umbrella to Uni. Obviously, it's useful to do so if it rains when you go back home, but it's annoying to carry around if it doesn't even rain. You look at the weather forecast, which hast three possible values: sunny, cloudy and rainy. 10 pt 10 min

1. Draw the decision network for bringing/leaving an umbrella depending on whether it does or doesn't rain later.
2. Explain *formally* how to compute whether or not to take an umbrella, assuming you know $P(rain = b|forecast = x)$ for all $b \in \texttt{Bool}, x \in \{\texttt{sunny}, \texttt{cloudy}, \texttt{rainy}\}$.

Solution:



Let $U_{\pm r,\pm u}$ be the base utilities of having an/no umbrella when it rains/doesn't rain. Assume the forecast says x, then compute:

$$U(umb) = P(rain = \top|forecast = x)U_{+r,+u} + P(rain = \bot|forecast = x)U_{-r,+u}$$
$$U(\neg umb) = P(rain = \top|forecast = x)U_{+r,-u} + P(rain = \bot|forecast = x)U_{-r,-u}$$

If the former is greater than the latter you should take an umbrella.

Problem 3.3 (Markov Decision Procedures)

12 pt

12 min

1. How do Markov decision procedures differ from (simple) decision networks?
2. How does the value iteration algorithm work? (Give an actual equation and explain its role in the algorithm)
3. What is the disadvantage of value iteration that is "fixed" by policy iteration?
4. How can we reduce *partially observable Markov decision procedures* to normal MDPs?

Solution:

1. In Markov decision procedures, the probabilistic model is a Markov Process (i.e. random variables are indexed over time, Markov Properties)
2. We assing a random utility to each state and update them using the Bellman equation:

$$U(s) = R(s) + \gamma \cdot \max_a \left(\sum_{s'} U(s') \cdot T(s, a, s') \right)$$

   Once this iteration has converged, we can compute the "best" action for each state by considering the expected utilities of all possible actions.

3. The policy resulting from value iteration can be stable long before the individual utilities have converged to their precise values.
4. By introducing belief states representing the probability distribution over the physical state space (i.e. the belief state space has one dimension for each physical state).

---

4 Markov Models

Problem 4.1 (Stock Market Predictions)
You bought SpaceY stock recently and try to predict whether to buy more or sell. The stock market is in one of two possible states; bull state or bear state. In a bull state, it will (in the long term) be advantageous to buy stock; in a bear state it will be more advantageous to sell. 10 pt 10 min

If the market is in a bull state, the probability it will still be in a bull state tomorrow is 60%. If it is in a bear state, the probability it will remain so tomorrow is 80%.

If the market is in a bull state, the probability that your stock will rise that day is 90%. If it is in a bear state, your stock will more likely fall (with 60% probability).

1. Explain what kind of probabilities *prediction*, *filtering* and *smoothing* compute in this scenario.
2. Give the underlying equations for the first two of these algorithms and explain what each variable in the equation represents.

Solution:

1. **Prediction** Given the *behavior* of the stock market up to time t_0, compute the probability of the state the stock market will be in at time $t_1 > t_0$

   Filtering Given the behavior of the stock market up to now, compute the probability of the state the stock market is in right now

   Smoothing Given the behavior of the stock market up to t_0, compute the probability that sta stock market was in some state at an earlier point $t_1 < t_0$.

2. $$P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \cdot \sum_{x_t} P(X_{t+1}|x_t, e_{1:t}) \cdot P(x_t|e_{1:t})$$

   Where X represents the state and e the behavior of the stock market.

---

5 Learning

Problem 5.1 (Home Decisions)

Eight people go sunbathing. Some of them got a sunburn, others didn't: 10 pt 10 min

Name	Hair	Height	Weight	Lotion	Result
Sarah	Blonde	Average	Light	No	Sunburned
Dana	Blonde	Tall	Average	Yes	None
Alex	Brown	Short	Average	Yes	None
Annie	Blonde	Short	Average	No	Sunburned
Julie	Blonde	Average	Light	No	None
Pete	Brown	Tall	Heavy	No	None
John	Brown	Average	Heavy	No	None
Ruth	Blonde	Average	Light	No	None

Explain how the information-theoretic decision tree learning algorithm would proceed on this table (up to two iterations). Explicitly state how to compute the information gain (and what that means).

Note that you do not need to compute any actual values; if it is helpful for your explanation, you may guess any values you might want to use.

Note that *Name* is only an index, not a (meaningful) attribute!

Solution:

$$E_0 := I(\langle\frac{2}{8},\frac{6}{8}\rangle) = -\frac{2}{8}\log_2(\frac{2}{8}) - \frac{6}{8}\log_2(\frac{6}{8}) \approx 0.81$$

$$\texttt{Gain(Hair)} = E_0 - \underbrace{\frac{5}{8}I(\langle\frac{2}{5},\frac{3}{5}\rangle)}_{\texttt{Blonde}} - \underbrace{\frac{3}{8}I(\langle 0,1\rangle)}_{\texttt{Brown}} \approx 0.20$$

$$\texttt{Gain(Height)} = E_0 - \underbrace{\frac{4}{8}I(\langle\frac{1}{4},\frac{3}{4}\rangle)}_{\texttt{Average}} - \underbrace{\frac{2}{8}I(\langle 0,1\rangle)}_{\texttt{Tall}} - \underbrace{\frac{2}{8}I(\langle\frac{1}{2},\frac{1}{2}\rangle)}_{\texttt{Short}} \approx 0.16$$

$$\texttt{Gain(Weight)} = E_0 - \underbrace{\frac{3}{8}I(\langle\frac{1}{3},\frac{2}{3}\rangle)}_{\texttt{Average}} - \underbrace{\frac{3}{8}I(\langle\frac{1}{3},\frac{2}{3}\rangle)}_{\texttt{Light}} - \underbrace{\frac{2}{8}I(\langle 0,1\rangle)}_{\texttt{Heavy}} \approx 0.12$$

$$\texttt{Gain(Lotion)} = E_0 - \underbrace{\frac{2}{8}I(\langle 0,1\rangle)}_{\texttt{Yes}} - \underbrace{\frac{6}{8}I(\langle\frac{2}{6},\frac{4}{6}\rangle)}_{\texttt{No}} \approx 0.12$$

`Hair` has the highest information gain, so we split here. All table entries with `Brown` have result `None`, so we continue with `Hair` = `Blonde`:

$$E_1 := I(\langle\frac{2}{5},\frac{3}{5}\rangle) \approx 0.97$$

$$\texttt{Gain}(\texttt{Height}) = E_1 - \underbrace{\frac{3}{5}I(\langle\frac{1}{3},\frac{2}{3}\rangle)}_{\texttt{Average}} - \underbrace{\frac{1}{5}I(\langle 0,1\rangle)}_{\texttt{Tall}} - \underbrace{\frac{1}{5}I(\langle 1,0\rangle)}_{\texttt{Short}} \qquad \approx 0.42$$

$$\texttt{Gain}(\texttt{Weight}) = E_1 - \underbrace{\frac{2}{5}I(\langle\frac{1}{2},\frac{1}{2}\rangle)}_{\texttt{Average}} - \underbrace{\frac{3}{5}I(\langle\frac{1}{3},\frac{2}{3}\rangle)}_{\texttt{Light}} - \underbrace{0}_{\texttt{Heavy}} \qquad \approx 0.02$$

$$\texttt{Gain}(\texttt{Lotion}) = E_1 - \underbrace{\frac{1}{5}I(\langle 0,1\rangle)}_{\texttt{Yes}} - \underbrace{\frac{4}{5}I(\langle\frac{2}{4},\frac{2}{4}\rangle)}_{\texttt{No}} \qquad \approx 0.17$$

`Height` has the highest information gain, so we proceed here. All short blondes are sunburned, all tall blondes are not, hence we only need consider `Average`...

Problem 5.2 (Backpropagation)

Explain what *Backpropagation* means in the context of Neural Networks, when and why we need it, and how to do it using an example. 8 pt 8 min

Solution:A possible answer:

Backpropagation is an algorithm for training feedforward neural networks for supervised learning. It computes the gradient of the loss function with respect to the weights of the network for a single input–output example.