

## Examination paper for TDT4171 Methods in AI

**Academic contact during examination:** Helge Langseth

**Phone:** 93015863

**Examination date:** 22/05-2019

**Examination time (from-to):** 15:00 – 19:00

**Permitted examination support material:** D. No printed or hand-written support material is allowed. A specific basic calculator is allowed.

**Other information:**

**Language:**

**Number of pages (front page excluded):** 5

**Number of pages enclosed:** 0

**Informasjon om trykking av eksamensoppgave**

**Originalen er:**

**1-sidig** ☐ **2-sidig** ☐

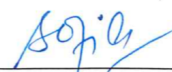
**sort/hvit** ☐ **farger** ☐

**skal ha flervalgskjema** ☐

**Checked by:**

05/09/2019

Date



Signature

## QUESTION 1: TRUE/FALSE (20 POINTS)

Question 1 can in total give 20 points. All sub-questions are to be answered by either “True” or “False”. No explanation is required, but if you feel the question is ill-posed you are of course allowed to add a justification to your answer. If you answer a sub-question correctly you get +2 points. A wrong answer gives -3 points. Not answering a sub-question results in 0 points. Nevertheless, please note that your total score from Question 1 will never be negative. A point total of zero will be carried forward if the sum of the scores from the sub-questions is negative.

- a) A Bayesian network model over the  $k$  Boolean variables  $X_1, X_2, \dots, X_k$  can represent every deterministic Boolean function over these  $k$  variables.

TRUE

- b) If we want to use Bayes formula we first have to make the Markov assumption.

FALSE. We do sometimes when building the model (e.g., for the Hidden Markov Models), but not in not-dynamic models. Also, the Markov-assumption is about the model-structure, while Bayes rule is about probability-manipulations.

- c) Case Based Reasoning utilizes a distance measure to compare a query to examples stored in the case-base.

TRUE. Some confusion here between the phrases “distance” and “similarity”, so beware if students answer FALSE and explain why. A statement like “FALSE. It uses similarity” therefore gives 2 points.

- d) If a system passes the Turing-test, it shows that the strong AI hypothesis is true.

FALSE

- e) The types of inference made by a Hidden Markov Model include *filtering* and *smoothing*.

TRUE

- f) Overfitting in machine learning means that a machine-learning model is fitted to capture both expert knowledge and information from data, while underfitting means that only the information from the data-set is utilized.

FALSE

- g) For every  $k > 3$  it holds that a decision tree with  $k$  internal nodes can express any Boolean function of  $k-1$  Boolean attributes.

FALSE

h) If A and B are independent random variables, then  $P(A=a, B=b) \leq P(A=a)$

TRUE.

i) Deep learning models can often contain more trainable parameters (“weights”) than the number of training examples used to learn the parameters.

TRUE, but the phrase “can often” was unfortunate, so be lenient if students give reasons for saying FALSE and say something about their reasoning.

j) Dropout is a regularization technique.

TRUE without argument is OK, but again a slightly unfortunate choice of words. Not knowing the phrase “regularization” but being able to explain what dropout means is OK.

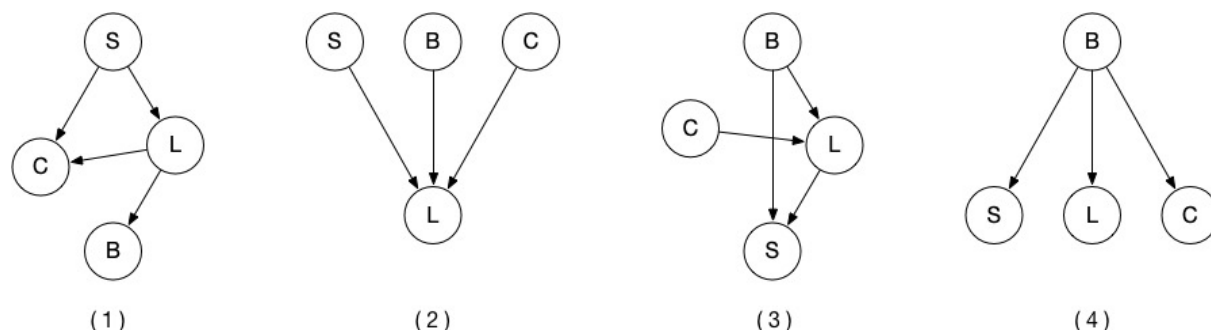
## QUESTION 2 – PROBABILISTIC REASONING (15 POINTS):

*Question 2 can give 15 points in total. Each sub-questions is weighted equally.*

Consider the Bayesian networks in the following figure. The model is used to examine medical patients. All patients that are examined are 60-year-olds, who are non-smokers or who has smoked for at least 40 years. The variables are all Boolean/binary, and interpreted as follows:

<b>S:</b> Smoking.	True if and only if a patient smokes.
<b>L:</b> LungCancer	True if and only if a patient has lung-cancer.
<b>C:</b> Cough	True if and only if the patient coughs significantly every morning
<b>B:</b> BiopsyTest	True if and only if the test result is positive for cancer

Note that none of the conditional probability tables (CPTs) are deterministic.



- a) Given current medical knowledge (or your understanding of it), which of the model structures are **most** correct?

**Number (1).**

All the others have, e.g., the link  $B \rightarrow L$  which makes no sense

- b) Which model has the fewest parameters? If there is a tie you should list all the “winners”.

**Number (4)** has 7 parameters ( $1+2+2+2$ ). **Number (1)** has  $1+2+4+2 = 9$ , so didn't win even if it was the most “natural” one.

- c) Write down a *reasonable* conditional probability table/CPT for the node **C** in Model (1). **Numbers are required.**

For example:  $P(c | s, l) = .95$ ,  $P(c | s, \text{not } l) = .8$ ,  $P(c | \text{not } s, l) = .2$ ,  $P(c | \text{not } s, \text{not } l) = .03$ ,  
And  $P(\text{not } c | \text{whatever}) = 1 - P(c | \text{same thing})$ .

Important here:

- The correct probability table is defined (that is:  $P(C | S, L)$  with S,L as parents)
- Probability for cough increases in both smoking and cancer.
- The probabilities must sum to 1 for each conditioning set if given (but no need to give both).
- The table is complete, that is, we need to have probability of cough for each combo of parents

Notational problems, like denoting the probability “P(C)” instead of “P(C | S, L)” is not a big issue.

- d) Using Model (1), derive a symbolic expression for the probability  $P(B=\text{True} \mid S=\text{False})$  in terms of the conditional probabilities available in the CPTs. **Do not plug in any numbers.** This means, find the equation so that  $P(B=\text{True} \mid S=\text{False})$  is expressed using `sum(s)` and `product(s)` of elements from the CPTs defined in this model.

Use the sum-rule to get rid of L. The idea is that

$P(B=\text{True} \mid S = \text{False}) = P(B=\text{True} \mid L=\text{True}, S = \text{False}) + P(B=\text{True} \mid L=\text{False}, S = \text{False})$   
This must be expressed using only probabilities from the model, so we get...

$$P(B=\text{True} \mid S = \text{False}) = \text{sum\_L } P(B=\text{True} \mid L=l) * P(L=l \mid S = \text{False})$$

$$= P(B=\text{True} \mid L=\text{True}) * P(L=\text{True} \mid S = \text{False}) \\ + P(B=\text{True} \mid L=\text{False}) * P(L=\text{False} \mid S = \text{False})$$

Note that very involved answers that could/should be simplified can still be correct, so take care here. A typical problem is that  $P(S = \text{False})$  is left in the calculations even without being required.

- e) Using Model (4), derive a symbolic expression for the probability  $P(B=\text{True} \mid S=\text{False})$  in terms of the conditional probabilities available in the CPTs. **Do not plug in any numbers.** This means, find the equation so that  $P(B=\text{True} \mid S=\text{False})$  is expressed using `sum(s)` and `product(s)` of elements from the CPTs defined in the model.

Bayes rule:

$$P(B=\text{True} \mid S = \text{False}) = P(S = \text{False} \mid B=\text{True}) * P(B=\text{True}) / P(S = \text{False})$$

=

$$P(S = \text{False} \mid B=\text{True}) * P(B=\text{True})$$

/

$$(P(S = \text{False} \mid B=\text{True}) * P(B=\text{True}) + P(S = \text{False} \mid B=\text{False}) * P(B=\text{False}))$$

Note that  $P(S=\text{False})$  is not part of the model (It is defined with B as parent), so leaving it at

$$P(S = \text{False} \mid B=\text{True}) * P(B=\text{True}) / P(S = \text{False})$$

is not fully correct, and does not give full score.

Claiming that  $P(B=\text{True} \mid S = \text{False}) = P(B=\text{True})$  obviously gives zero points.

### QUESTION 3 – PROBABILISTIC MODELLING (10 POINTS):

Question 3 can give 10 points in total: 4 points from sub-question (a), 4 points from sub-question (b), and 2 points from sub-question (c).

Peter wakes up one morning with a sore throat. It may be the beginning of a cold or he may suffer from an angina, as both diseases can cause a sore throat. An angina also causes yellow spots down the throat.

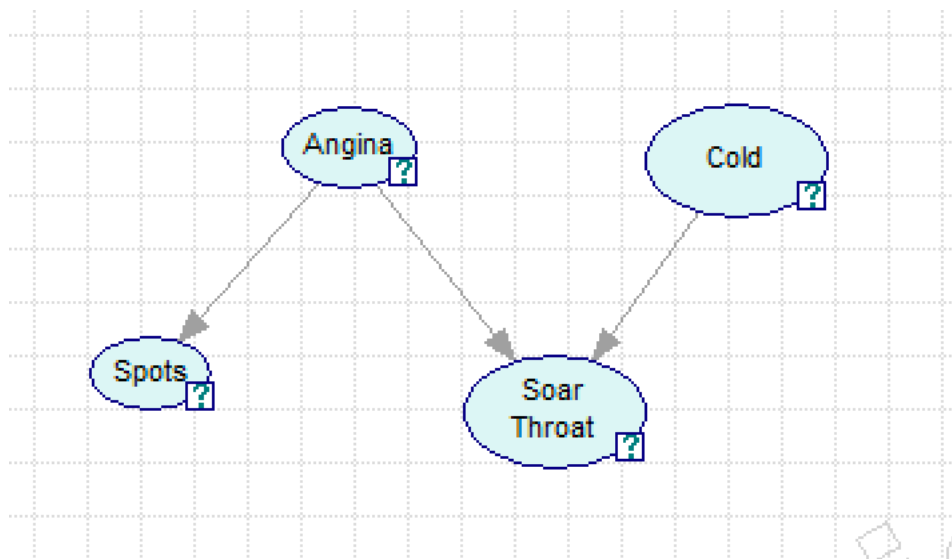
- a) Build a Bayesian network model to represent the description above. Make sure to show explicitly which variables are included in the domain, and list the states for each variable. You need to show the *network structure*, but you will **not** need to show the conditional probability tables.

Variables: Angina, Cold, Spots, SoreThroat. All variables are Boolean. All arrows must be in the causal direction. Statements like “YellowSpots points to Angina because this is how we reason” makes non-causal model even worse (Use Bayes’ rule to find out what you want).

**Failing to mention state-spaces (e.g., all binary) costs one point.**

If S is not part of the model (because it was observed True already), the model becomes much more difficult to create. One would need to somehow connect Angina and Cold, and how to do that is unclear. So, SoreThroat should be there.

Combining Angina and Cold into “Sickness” with states “Angina” and “Cold” is suboptimal. The correct model does not assume that exactly one disease is present (as this was not mentioned in the text).



- b) Do you think that Bayesian Networks constitute a natural modelling framework for this problem domain? How can one characterize problem domains where Bayesian networks can be used with success? Can you give an example of a problem domain where Bayesian networks do not fit well?

I think **yes**.

- Clearly defined variables with well-defined semantics.
- Natural causal story leading to the structure.
- Uncertainty is present
- We can populate the CPTs, either because we have access to a domain expert (including ourselves) or data to learn from.
- While the model is simplified, e.g., there may be other reasons for the sore throat, this is not a “BN problem”. In fact, we are free to capture this in the probability tables by saying that  $P(S=True \mid A=FALSE, C=FALSE) > 0$ .

**Causality** is the key concept that must be here.

Situations where BNs do not fit well: Something that does not have the properties listed above. Answers like “models with many variables” are not good. This is a feature of the **problem**, not the **modelling framework**. Same goes for comments about need to have many (conditional) independencies: If the model is with very many links it is because the problem is hard, not because BNs do not fit. (We gain less from using them, but will never lose from employing BNs compared to an analysis of the full joint.) Comments about time and BN vs Hidden Markov are also so-so, in that a HMM **can** be understood as a Bayes net, too. Here we’ve nevertheless been lenient. Also, BNs **can** handle continuous variables using some adaptations, so that is not a good answer. Still, we have not talked about continuous variables in the course, hence haven’t punished these answers either.

Example of bad fit: For instance, a system that classifies images of cats and dogs from pixel-information. We understand that there are statistical dependencies at the pixel level, but it is hard to capture this by a model (in particular if we want the model to be causal), and hard to make reasonable CPTs. Any reasonable answer will obviously give full score.

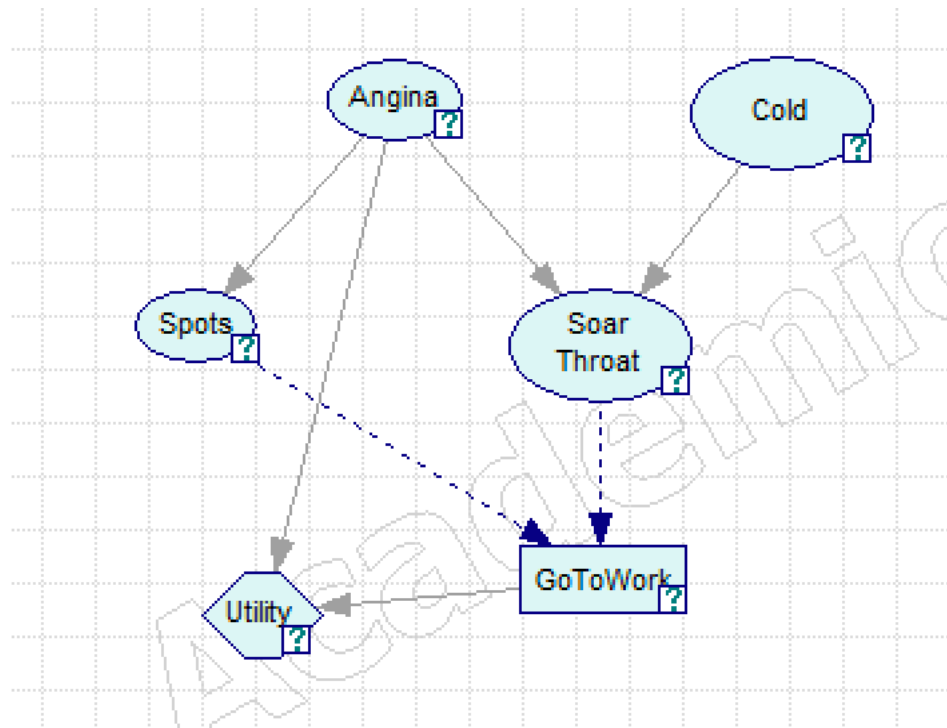
*If Peter has a cold and not the angina he will go to work. The reason for this is that if he stays at home with only a cold his co-workers may disrespect him and think he is not pulling his weight at the office. On the other hand, he should stay at home if he has the more severe condition of angina.*

- c) Extend the model from part (a) to model the decision problem “Should Peter stay at home or should he go to work?”. Make sure that your final model is a complete decision network (also known as influence diagram). You do **not** have to add the quantitative part to the model.

I like the model structure below, but I am willing to accept other reasonable models, too. The key elements of the model are:

- We need a decision node and a utility node
- We need the same BN structure as before
- The utility node must have either Angina or Cold (or both) as parent, and must have the decision as parent, too.
- The utility node has no children.
- The decision-node can only have the utility as child. If it points to other variables it means that variable is (potentially) changed by the decision, and that was not the case in this story.
- Spots and SoreThroat are both observed before the decision is made. Therefore we should have links from these variables to the decision-node. I realize this may be a bit beyond the average student’s level, though, so we should not punish the lack of these dashed links.

- Fancy stuff like adding a node for the response from co-workers is OK, and does not count either way. It is OK to have more than one utility node if one says that the sum is the target, but it is **not** correct to have more than one decision node (e.g., one for “StayHome”, and another for “GoToWork”) because this does not respect that one cannot be both places at the same time.





#### QUESTION 4 – DECISION TREE LEARNING (15 POINTS)

*Each sub-question is given the same weight (5 points) for Question 4.*

Below you are given a table of training examples. The training-set contains 10 examples in total. Each example is described using the two attributes X1 and X2, as well as the class C. X1 and X2 are Boolean, while the class is either “+” or “-”. We will use this data-set to build a decision-tree.

Training Example	X1	X2	Class
D1	0	0	+
D2	0	0	+
D3	0	0	+
D4	0	0	+
D5	0	1	-
D6	0	1	-
D7	0	1	-
D8	1	0	-
D9	1	1	+
D10	1	1	+

While solving Sub-question c) you can use that  $I(p, n)$ , the entropy / “information need” for a dataset containing  $p$  positive and  $n$  negative examples, is given by this formula:

$$I(p, n) = -\frac{p}{p+n} \log_2 \left( \frac{p}{p+n} \right) - \frac{n}{p+n} \log_2 \left( \frac{n}{p+n} \right)$$

Furthermore, to help with the calculations in that sub-question, you can use the table below to evaluate  $I(p, n)$ . For given  $p$  and  $n$ , calculate  $p/(p+n)$ , and find the entry in the table closest to that decimal number (e.g., if you want to calculate the entropy of a dataset with  $p=1, n=2$  you calculate  $p/(p+n)=1/(1+2)=1/3 \approx 0.3$ , so should look up the entry for 0.3). The table then gives the entropy (0.9 for the example). It is sufficient to make the calculations in sub-question c) rounded to one digit behind the decimal point.

$p/(p+n)$	$I(p, n)$
0,0	0,0
0,1	0,5
0,2	0,7
0,3	0,9
0,4	1,0
0,5	1,0
0,6	1,0
0,7	0,9
0,8	0,7
0,9	0,5
1,0	0,0

- a) Define the a decision tree: What is it used for, what does it look like, what is the purpose of the internal nodes, and what happens at the leaf-nodes?

A decision tree is used for classification or regression. It is a tree-structured graph containing two types of nodes: internal nodes/decision nodes and leaf-nodes/classification nodes. It is used when queries given to the model is of the (attribute – value)-type, meaning each object is represented by a list of attribute-values, and where the set of attributes is shared by all the objects.

The internal nodes are used as decisions: It represents one of the attributes in the dataset (here  $X_1$  or  $X_2$ ), and has outgoing arcs labelled with the possible values the variable can take (here TRUE or FALSE for all internal nodes).

The Leaf-nodes are classification nodes: When you arrive at a leaf it tells you what class to choose for that example, so it selects between + and – in this example.

The DT works by getting a query in “at the top”, checking the attribute mentioned in the top-node, and selecting the outgoing edge/branch that fits with the value of the query. If the branch leads to a new internal node, the process of checking the query top choose an outgoing arc is repeated. Eventually we get to a leaf, where the “end result” (classification or – not covered in the course – parameterization for an ML technique like regression parameters) are returned. In a way we can understand it as a collection of rules, where we have IF (Attribute1==Value1) AND (Attribute2==Value2) AND ... THEN CLASS= Class1.

There is no need to talk about learning a decision tree in this sub-question, and all this about splitting a dataset and whatnot is not important or relevant here. Rather, we want the syntax and semantics clearly described.

- b) Give a high-level description of the decision-tree learning algorithm we used in the course. One way to do this is to give the algorithm as pseudo-code.

Start with the full dataset, and all attributes available.

1. If all elements of the dataset have the same class, then make a leaf-node with that class, and return
2. If there are no attributes left, then make a leaf-node with the class defined by the major-ity class in the dataset, and return.
3. Pick a «best possible» decision-node. The node is chosen among the possible attributes, and which one is chosen depends on the data supplied.
4. Create one output-link per possible state of the decision variable, and for that output link, create a new decision-tree (run the algo recursively) with data constrained to the subset of the data where the chosen attribute has this label and attribute-set given by all attributes except the chosen one.

Key issues here:

- The recursive behavior of the algorithm.
- That each “best”-node generates sub-trees according to the states/values it can take.
- How data is “split” into subsets and passed along each branch/arc leading out of the node before the recursive call is made.

- c) Use the decision-tree learning algorithm you just described to find a decision-tree for the given dataset. In particular, show which attribute is used at the root, and why. Give numerical results wherever relevant. Show which training-examples are involved where. Describe clearly what is encoded at the leaf-nodes.

The key here is to choose the variables in the right order when building the tree. The way to choose is to look at information gain. That is, we choose the variable that reduces the *expected* remaining entropy the most. Expectation is over the branches leading out of the new node, where the entropies are weighted according to how many data-samples we see for each branch. To not aggregate over the outcomes correctly (e.g, use an entropy for one of the states as the representative for the attribute, or to sum over states without using weights) is wrong.

### Entropy at the beginning:

Use all ten examples. 6 positives, 4 negatives, so we use  $I(6,4) = 1.0$  (This value is not really needed for the calculations to work, so no harm if it isn't there and the X2-at-root conclusion is correct)

### If we start with X1 at the root:

- For  $X1=0$ : 7 examples (D1, D2, D3, D4, D5, D6, D7),  $p=4, n=3$ .  $I(4,3)=1.0$
- For  $X1=1$ : 3 examples (D8, D9, D10),  $p=2, n=1$ .  $I(2,1)=0.9$
- Expected remaining entropy:  $7/10 \cdot 1.0 + 3/10 \cdot 0.9 = 0.97 \approx 1.0$
- Information Gain: Start-entropy – expected remaining entropy =  $1.0 - 1.0 = 0.0$

### If we start with X2 at the root:

- For  $X2=0$ : 5 examples (D1, D2, D3, D4, D8),  $p=4, n=1$ .  $I(4,1)=0.7$
- For  $X2=1$ : 5 examples (D5, D6, D7, D9, D10),  $p=2, n=3$ .  $I(2,3)=1.0$
- Expected remaining entropy:  $5/10 \cdot 0.7 + 5/10 \cdot 1.0 = 0.85 \approx 0.9$

Information Gain: Start-entropy – expected remaining entropy =  $1.0 - 0.9 = 0.1$

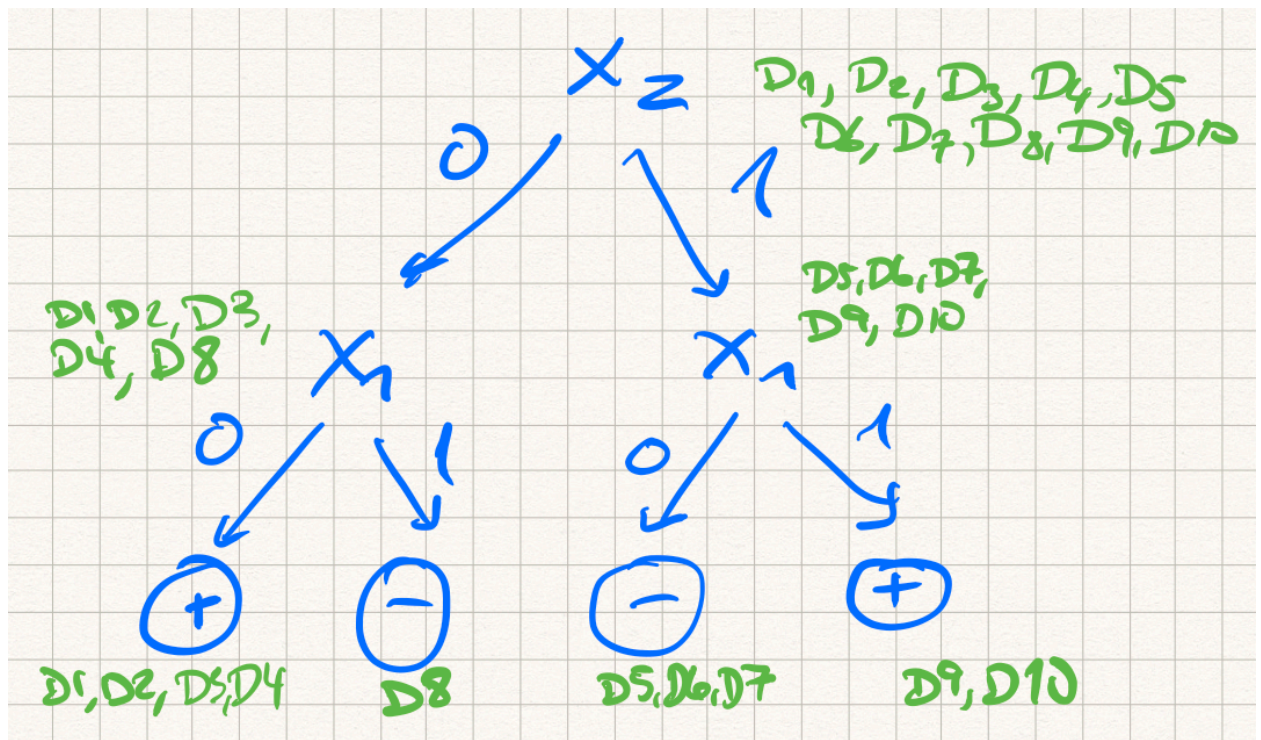
**Therefore, start with X2 at root.**

### Look at leaf with X2=0:

- We have 5 examples (D1, D2, D3, D4, D8),  $p=4, n=1$ .
- Since we have remaining information, consider X1 (the only remaining variable).
- $X1 = 0$  gives  $p=4, n=0$  so entropy 0;  $X1 = 1$  gives  $p=0, n=1$  so entropy = 0. Expected remaining entropy is 0, less than the initial entropy so include X1 as a new internal node.
- This new node results in pure subsets, so we add leaf-nodes with the correct classes.
- **The checking of X1 reduces entropy may be missing without point penalty.**

### Look at leaf with X2=1:

- We have 5 examples (D5, D6, D7, D9, D10),  $p=2, n=3$ .
- Again, X1 is the only candidate. Check if it improves the model to include it as above. It does.
- $X1 = 0$ : three examples (D5, D6, D7), with  $p=0, n=3$ . Add a leaf-node with class “-”
- $X1 = 1$ : three examples (D9, D10), with  $p=2, n=0$ . So, add a leaf-node with class “+”



---

Students will find the examination results in Studentweb. Please contact the department if you have questions about your results. The Examinations Office will not be able to answer this.

### QUESTION 5: MIXED QUESTIONS (20 POINTS)

The maximum points for this question is 20 points, each sub-question is given equal weight.

a) **Briefly** explain the four steps of the CBR cycle.

**Retrieve:** Find a case/set of cases most similar/least distant from the query/problem. Clever stuff on how it works not required, but need mentioning of a similarity measure for full score.

**Reuse:** If applicable, fix the solution of the retrieved case(s) so that it is/they are applicable in the setting of the problem-case. Again, no clever stuff needed.

**Revise:** An external step, where the solution is “OK’ed” by a domain expert or where a failed solution can be fixed if it didn’t work.

**Retain:** Save the solved case in the case-base. Clever stuff like updating index structures etc not required.

b) You are invited to participate in the following game, where you can play only once:

*You throw a pair of dice once. If both dice show the same number of dots, you win a dollar-amount equal to the number of dots on one die (so a pair of 1’s wins you \$1, a pair of 2’s wins you \$2, and so on). If you do not get a pair you win nothing. You will have to pay \$1 to partake.*

- What does it mean that an agent is *rational*?
- Is it rational of you to play the game if your utility for money is linear in the amount you have?
- Can you create a utility function that changes what is rational behavior?

Rational (short answer): Use the available information to maximize your expected utility. Key to talk about **expected** and **utility**. An answer like “Do what is best in your situation” is not sufficient. **One point for this one.**

It is not rational to play the game. A player can expect to win in 6 of 36 possible settings, and the expected payback is  $1/36 (1 + 2 + 3 + 4 + 5 + 6) = 21/36$ . Since I always pay to partake, my expected winnings is  $21/36 - 1 = -15/36$  or about -.42. I expect to have less money than when I started, a decrease in expected utility, so not going there. Statements like “You lose more often than you win, so I don’t want to join” are not sufficient for full score. The utility-function must be involved! **Two points for this one if done correctly.**

Any utility function that can be shown to give the “yes please” outcome should be accepted. People can be “clever” and say that the utility of playing a game for them is worth \$100, or keep looking only at the money, and for instance say that they have no need for any amount below \$3, so that  $U(\$k) = 0$  for  $k < 3$ , while  $U(\$k) = k$  otherwise. The point here is that an agent is free to choose his/her own utility function, thus changing the game (requiring cheaper entry-fee) or loading the dice are not OK answers. **Two points here, too**

- c) **Briefly** describe reinforcement learning, how it differs from the other learning algorithms in this course, and give an example of where reinforcement learning can be used.

**Any description here that makes sense works, but some key points may be:**

- Used when we for different “states”  $s$  are asked to make a decision/choose an action  $a$ . This process is repeated (typically many) times. As we move along we may with irregular intervals get feedback in terms of reward. The **sequential** nature of RL-type settings is key to get full score.
- RL being different from supervised learning because we do not get a “correct answer” for each question given to the system, we only get a result at the end of an episode. Say we want to learn chess. Supervised learning requires labelled examples (like  $x$ =board-state and  $f(x)$ =best move), that we do not have here.
- It is not a complete answer to say that it is like supervised learning, only that we in RL get results “every now and then” instead of all the time. This would mean that RL is semi-supervised learning, something it is not. Rather, we here **must remember the sequential aspect** of the domain (many actions are chosen).
- A description of the learning algorithm is a plus.
- Showing a bit of understanding in terms of mentioning Q-learning (the only RL technique we did in detail), talking about  $Q(s,a)$  and in particular **why** we need to aggregate information of aggregated discounted rewards at this level (and not at state-level as we do for MDPs) is a big plus. Uncertainty in the domain and the unknown effect of actions is the key here.
- The explore/exploit tradeoff should be mentioned.
- Nice to tie RL to, e.g., playing chess (where we only get to know if a sequence of moves eventually was clever enough to beat the opponent, not what to do for each board-state).
- Typically, RL is not started with a given “dataset”. Rather, experience is gathered through interacting in an environment (or a simulator). This generates data/experience to learn from, though, so the learning is based on experience.
- **Lacking example of RL: Loose one point**

- d) **Briefly** describe the most important properties of deep learning.

Again, any description here that makes sense works. Some key points may be:

- DL uses layers of hidden variables (perceptrons) to capture increasingly complex patterns in the data, eventually ending at an output layer (e.g., a classification)
- Each node/perceptron is doing very simple calculations, and is itself unable to “do much”, it is the combination of nodes in layers that captures the interesting properties.
- Extends “classical neural nets” in that one chooses to go deeper than before, typically using fewer nodes per hidden layer.
- Special setups for special data, like convolutions (images, other local structure) and recurrence (sequences).
- Learning using backprop (or extensions thereof)
- Overfitting problems, regularizations, dedicated loss-functions.