Department of Computer Science

Examination paper for TDT4171 Artificial Intelligence Methods

Academic contact during examination: Odd Erik Gundersen

Phone: 47 63 70 75

Examination date: June 4 2018

Examination time (from-to): 0900 - 1300

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

Other information:

Language: English

Number of pages (front page excluded): 4

Number of pages enclosed: 0

Informasjon om trykking av eksamensoppgave
Originalen er:
1-sidig □ **2-sidig** □
sort/hvit □ **farger** □
skal ha flervalgskjema □

Checked by:

Date Signature

Task 1: True/False (10p)

For each correct answer you will get 2p, and for each wrong answer you get -1. The lowest score you can get on Task 1 is 0.

a) **Uncertainty (2p):** The marginal probability of a variable can be found by summing out the probabilities for each possible value of the other variables:

$$\boldsymbol{P}(\boldsymbol{Y}) = \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \boldsymbol{P}(\boldsymbol{Y}, \boldsymbol{z})$$

b) **Case-based reasoning (2p):** The CBR cycle has four steps: 1) Retrieve, 2) Reuse, 3) Revise and 4) Retain.

c) **Cross-validation (2p):** k-fold cross-validation is the same as leave-one-out cross-validation where k = n and n is the number of test examples.

d) **Deep learning (2p):** Dropout is a way to avoid overfitting.

e) **Deep learning (2p):** When training with backprop, ReLU typically learns much faster in networks with many layers, allowing training of a deep supervised network without unsupervised pre-training.

Task 2: Naive Bayes - spam or ham (15p)

a) **Assumption (5p):** Describe the assumption that is underlying the naive Bayes model.

b) **Probability distribution (5p):** Write the joint probability distribution for the naive Bayes model P($Cause$, $Effect_1$, ..., $Effect_n$):

c) **Spam classification (5p):** Compute the probability distribution for the *spam mail training set* and use the probabilities to classify the test set.

(Hint: Spam if **P**($Spam$, $Effect_1$, ..., $Effect_i$) > **P**(Ham, $Effect_1$, ..., $Effect_i$))

Table 1: Training set - spam or ham.

ID	Spam	Words in email				
		prince	**nigeria**	**money**	**travel**	**office**
1	y	1	1	1	0	0
2	n	1	0	1	1	0
3	y	0	1	1	0	0
4	n	0	1	0	1	1
5	n	0	0	1	1	1
6	y	1	1	1	1	1
7	n	0	0	1	1	0
8	n	0	0	0	0	1
9	y	1	1	0	0	0
10	y	1	1	1	1	0

Table 2: The test set.

ID	Spam	Words in email				
		prince	**nigeria**	**money**	**travel**	**office**
11	?	1	0	1	1	1

Task 3: Hidden Markov Models (25p)

a) **Markov process description (4p):** Given the short description below, draw the Hidden Markov Model and set up all the conditional distribution tables describing the example (3p). Which order does this Markov process have assuming that the description is complete, and all the dependencies are stated in the description (1p)?

I have found that there is a relationship between whether the light is on in my neighbor's house and whether he is home. If the light is on there is a 0.7 probability that he is home, but if he is away there is still 0.3 probability that the light is on. The probability that my neighbor is home today is 0.8 if he was home yesterday. If he was away yesterday the probability is 0.2 that he is home today. When I first moved in, I thought there was a 0.5 probability that he was home.

b) **Markov models (4p):** Explain the Markov assumption (2p). Explain what a stationary Markov process is (2p).

b) **Inference (10p):** Give a short explanation of the basic inference tasks for temporal models.

c) **State estimation (10p):** Compute the probability $P(H_2 | l_1, l_2)$ the model described in a), that is whether the neighbor is home on day two given that the lights were on both day one and day two.

d) **Mixing time (2p):** What is mixing time?

Task 4: Making complex decisions

a) **Algorithm (5p):** Describe the value iteration algorithm.

b) **Example (10p):** Execute the value iteration algorithm (two iterations) on the following example where a robot is to navigate on top of a mountain to send signals to its friends in the outer space (10), but with a steep cliff (-7) and some muddy ground (-2):

	1	2	3
1			
2		10	-7
3			-2

The robot uses battery for each grid cell it moves at a cost of -0.2. The robot that can move north, south, east and west with a probability of 0.8 going in the wanted direction and 0.2 probability of going sideways (0.1 for each direction). If it moves into a wall it effectively stands still.

The answer should be twofold: 1) a grid with the resulting utilities and 2) the optimal policy.

Task 5: Decision Trees (25p)

You are hired by the local power company to predict the power production of their wind farm. You are given the data set listed in table 3.

Table 3: Wind power prediction training data set from the local power company.

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
1	H	N	H	C	H
2	L	N	H	C	L
3	M	S	H	S	H
4	L	S	L	S	L
5	M	S	L	C	L
6	H	N	H	C	H
7	M	N	H	S	H
8	M	N	H	C	H
9	H	S	L	S	H
10	M	S	H	S	L

The attributes include the ID for the sample data (*ID*), the wind farm's production for the previous hour (*prod_prev_hour*), the current wind direction at the windfarm (*wind_direction*), the forecasted wind strength the next hour (*pred_wind_strength*), the forecasted weather (*weather*) and the actual production for the given hour (*prod*).

Important: The samples in the data set are not consecutive, but randomized, and are sampled from a larger data set.

You decide that you can use decision trees to predict the power production of the wind farm. Good decision!

a) **Information gain (5p):** Explain information gain and how it can be used to select attributes for a decision tree. Give an example by using one of the attributes from the data set in table 3.

b) **Decision tree (12p):** Use information gain to generate a decision tree based on the provided wind power prediction training data set. The answer should be a drawing of the decision tree.

c) **Evaluation (3p):** The local power company wants to test your prediction algorithm and provides some test data listed in table 4.

Table 4: Wind power prediction test set.

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
11	M	S	H	S	M

Run your decision tree on the data. Show the execution path in your decision tree. What is your prediction?

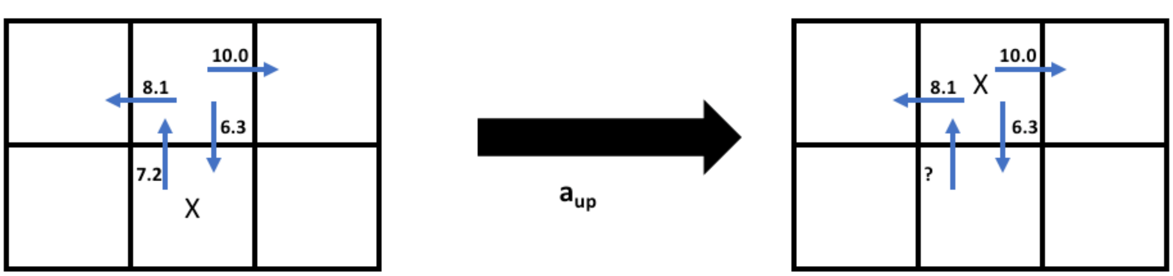
d) **Advice (5p):** The local power company is not happy with your result, but you explain to them that the experiment has some flaws. Please advise them on how to improve the experiment.

Task 6: Reinforcement learning

a) **Q-learning algorithm (5p):** Give a short description of the Q-learning algorithm and provide pseudo code for it.

b) **Example (5p):** An agent designated X in the drawing below is located in state S_1 (bottom, middle) is doing Q-learning with fuel expenditure of 0.1 and a discount factor of 0.9. X performs an action a_{up} and ends up in S_2 (top, middle):

Use this information to update $Q(a_{up}, S_1)$.



Task 1: True/False (10p)

For each correct answer you will get 2p, and for each wrong answer you get -1. The lowest score you can get on Task 1 is 0.

a) **Uncertainty (2p):** The marginal probability of a variable can be found by summing out the probabilities for each possible value of the other variables:

$$\boldsymbol{P}(\boldsymbol{Y}) = \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \boldsymbol{P}(\boldsymbol{Y}, \boldsymbol{z})$$

Solution: True

b) **Case-based reasoning (2p):** The CBR cycle has four steps: 1) Retrieve, 2) Reuse, 3) Revise and 4) Retain.

Solution: True

c) **Cross-validation (2p):** k-fold cross-validation is the same as leave-one-out cross-validation where k = n and n is the number of test examples.

Solution: False - n is the number of training examples.

d) **Deep learning (2p):** Dropout is a way to avoid overfitting.

Solution: True

e) **Deep learning (2p):** When training 4 hidden layers of logistics units with backprop, the first hidden layer is training 100 times slower than the last one.

Solution: True

Task 2: Naive Bayes - spam or ham (15p)

a) **Assumption (5p):** Describe the assumption that is underlying the naive Bayes model.

Solution:

Conditional independence: A single cause influences a number of effects, all of which are conditionally independent. The effect variables are not actually conditionally independent given the cause variable, but naive Bayes models can work surprisingly well even when the conditional independence assumption is not true.

b) **Probability distribution (5p):** Write the joint probability distribution for the naive Bayes model P(*Cause*, *Effect*$_1$, ..., *Effect*$_n$):

Solution:

$$P(Cause, Effect_1, \ldots, Effect_n) = P(Cause) \prod_i P(Effect_i|Cause)$$

c) **Spam classification (5p):** Compute the probability distribution for the *spam mail training set* and use the probabilities to classify the test set.

(Hint: Spam if **P**(*Spam*, *Effect₁*, ..., *Effectᵢ*) > **P**(Ham, *Effect₁*, ..., *Effectᵢ*))

Table 1: Training set - spam or ham.

ID	Spam	Words in email				
		prince	**nigeria**	**money**	**travel**	**office**
1	y	1	1	1	0	0
2	n	1	0	1	1	0
3	y	0	1	1	0	0
4	n	0	1	0	1	1
5	n	0	0	1	1	1
6	y	1	1	1	1	1
7	n	0	0	1	1	0
8	n	0	0	0	0	1
9	y	1	1	0	0	0
10	y	1	1	1	1	0

Table 2: The test set.

ID	Spam	Words in email				
		prince	**nigeria**	**money**	**travel**	**office**
11	?	1	0	1	1	1

Solution: Probability distributions

ID	Spam	prince	nigeria	money	travel	office
1	y	1	1	1	0	0
3	y	0	1	1	0	0
6	y	1	1	1	1	1
9	y	1	1	0	0	0
10	y	1	1	1	1	0
	P(word|spam)	0,8	1	0,8	0,4	0,2
2	n	1	0	1	1	0
4	n	0	1	0	1	1
5	n	0	0	1	1	1
7	n	0	0	1	1	0
8	n	0	0	0	0	1
	P(word|ham)	0,2	0,2	0,6	0,8	0,6

Solution: Classification

	prince	nigeria	money	travel	office
Words in email	1		1	1	1
P(spam)	0,8		0,8	0,4	0,2
P(ham)	0,2		0,6	0,8	0,6

$\mathrm{P}(spam, prince, \mathrm{money}, travel, office) =$
$P(spam)P(prince|spam)P(money|spam)P(travel|spam)P(office|spam)$=0,5*0,8*0,8*0,4*0,2=0,0256

$\mathrm{P}(ham, prince, \mathrm{money}, travel, office) =$
$P(ham)P(prince|ham)P(money|ham)P(travel|ham)P(office|ham)$=0,5*0,2*0,6*0,8*0,6=0,0288

$$\mathrm{P}(ham, prince, \mathrm{money}, travel, office) > \mathrm{P}(spam, prince, \mathrm{money}, travel, office)$$

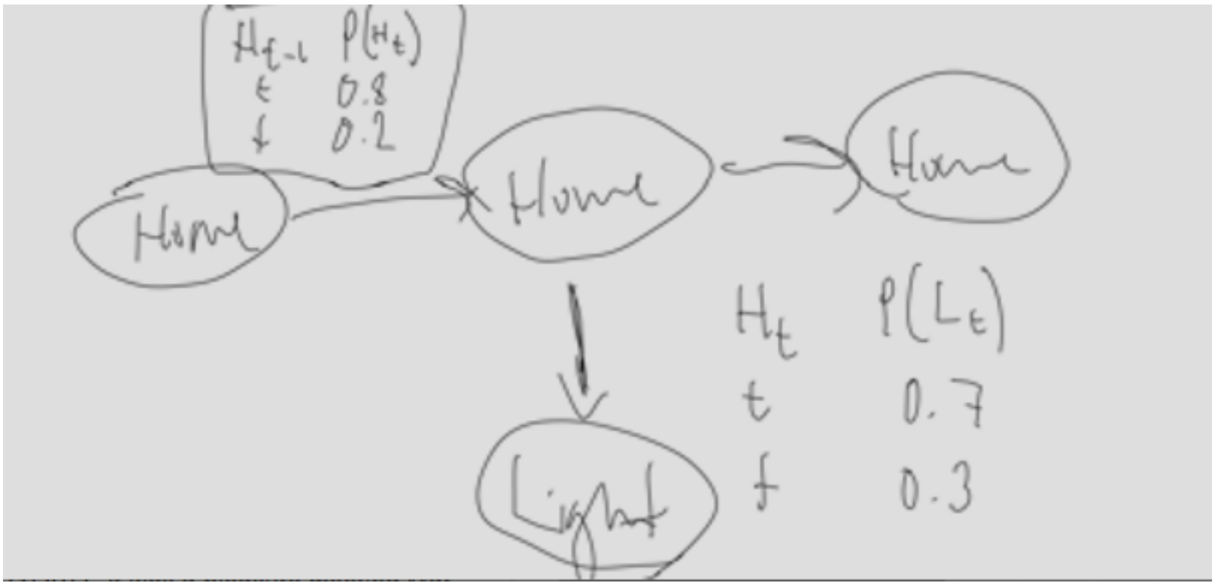
Email is ham.

Task 3: Hidden Markov Models (25p)
a) **Model process (4p):** Given the short description below, draw the Hidden Markov Model and set up all the conditional distributions describing the example (3p). Which order does this Markov process have (1p)?

I have found that there is a relationship between whether the light is on in my neighbor's house and whether he is home. If the light is on there is a 0.7 probability that he is home, but if he is away there is still 0.3 probability that the light is on. The probability that my neighbor is home today is 0.8 if he was home yesterday. If he was away yesterday the probability is 0.2 that he is home today. When I first moved in, I thought there was a 0.5 probability that he was home.

Note: " *If the light is on there is a 0.7 probability that he is home, but if he is away there is still 0.3 probability that the light is on." is meant as "If the light is on there is a 0.7 probability that he is home, but there is still a 0.3 probability that the light is on if he is away."*

Hidden Markov model:



The prior is: $P(H_0)$ = <0.5, 0.5>

Order: This is a first order Markov process.

b) **Markov models (4p):** Explain the Markov assumption (2p). Explain what a stationary Markov process is (2p).

Markov assumption: the current state depends only on a finite fixed number of previous states.

$$P(X_t|X_{0:t-1}) = P(X_t|X_{t-n})$$

where n < t-1

Stationary Markov process: Is a process of change that is governed by laws that do not themselves changes. This means that the transition model is stationary and not dynamic.

b) **Inference (10p):** Give a short explanation of the basic inference tasks for temporal models.

This is the level of description that is expected:

Having set up the structure of a generic temporal model, we can formulate the basic inference tasks that must be solved:

- **Filtering**: This is the task of computing the **belief state**—the posterior distribution over the most recent state—given all evidence to date. Filtering[2] is also called **state estimation**. In our example, we wish to compute $\mathbf{P}(\mathbf{X}_t \mid \mathbf{e}_{1:t})$. In the umbrella example, this would mean computing the probability of rain today, given all the observations of the umbrella carrier made so far. Filtering is what a rational agent does to keep track of the current state so that rational decisions can be made. It turns out that an almost identical calculation provides the likelihood of the evidence sequence, $P(\mathbf{e}_{1:t})$.
- **Prediction**: This is the task of computing the posterior distribution over the *future* state, given all evidence to date. That is, we wish to compute $\mathbf{P}(\mathbf{X}_{t+k} \mid \mathbf{e}_{1:t})$ for some $k > 0$. In the umbrella example, this might mean computing the probability of rain three days from now, given all the observations to date. Prediction is useful for evaluating possible courses of action based on their expected outcomes.

[2] The term "filtering" refers to the roots of this problem in early work on signal processing, where the problem is to filter out the noise in a signal by estimating its underlying properties.

- **Smoothing**: This is the task of computing the posterior distribution over a *past* state, given all evidence up to the present. That is, we wish to compute $\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:t})$ for some k such that $0 \leq k < t$. In the umbrella example, it might mean computing the probability that it rained last Wednesday, given all the observations of the umbrella carrier made up to today. Smoothing provides a better estimate of the state than was available at the time, because it incorporates more evidence.[3]
- **Most likely explanation**: Given a sequence of observations, we might wish to find the sequence of states that is most likely to have generated those observations. That is, we wish to compute $\operatorname{argmax}_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t} \mid \mathbf{e}_{1:t})$. For example, if the umbrella appears on each of the first three days and is absent on the fourth, then the most likely explanation is that it rained on the first three days and did not rain on the fourth. Algorithms for this task are useful in many applications, including speech recognition—where the aim is to find the most likely sequence of words, given a series of sounds—and the reconstruction of bit strings transmitted over a noisy channel.

c) **State estimation (10p):** Estimate the state for two steps of the given model neighbor described in a). That is compute the probability $P(H_2|l_1,l_2)$, that is whether the neighbor is home on day two given that the lights were on both day one and day two.

P(H0) = <0.5, 0.5>

$P(H_1) = \sum_{h_0} P(H_1|h_0)P(h_0)$ = <0.8,0.2>*0.5 + <0.2,0.8>*0.5 = <0.5, 0.5>

$P(H_1|l_1) = \alpha P(l_1|H_1)P(H_1) = \alpha <0.7,0.3><0.5,0.5> = \alpha <0.35,0.15> = <0.7,0.3>$

$P(H_2|l_1) = \sum_{h_1} P(H_2|h_1)P(h_1|l_1) =< 0.7{,}0.3 >* 0.7 +< 0.3{,}0.7 >* 0.3 =< 0.49{,}0.21 > +< 0.09{,}0.21 >=<0.58{,}0.42>$

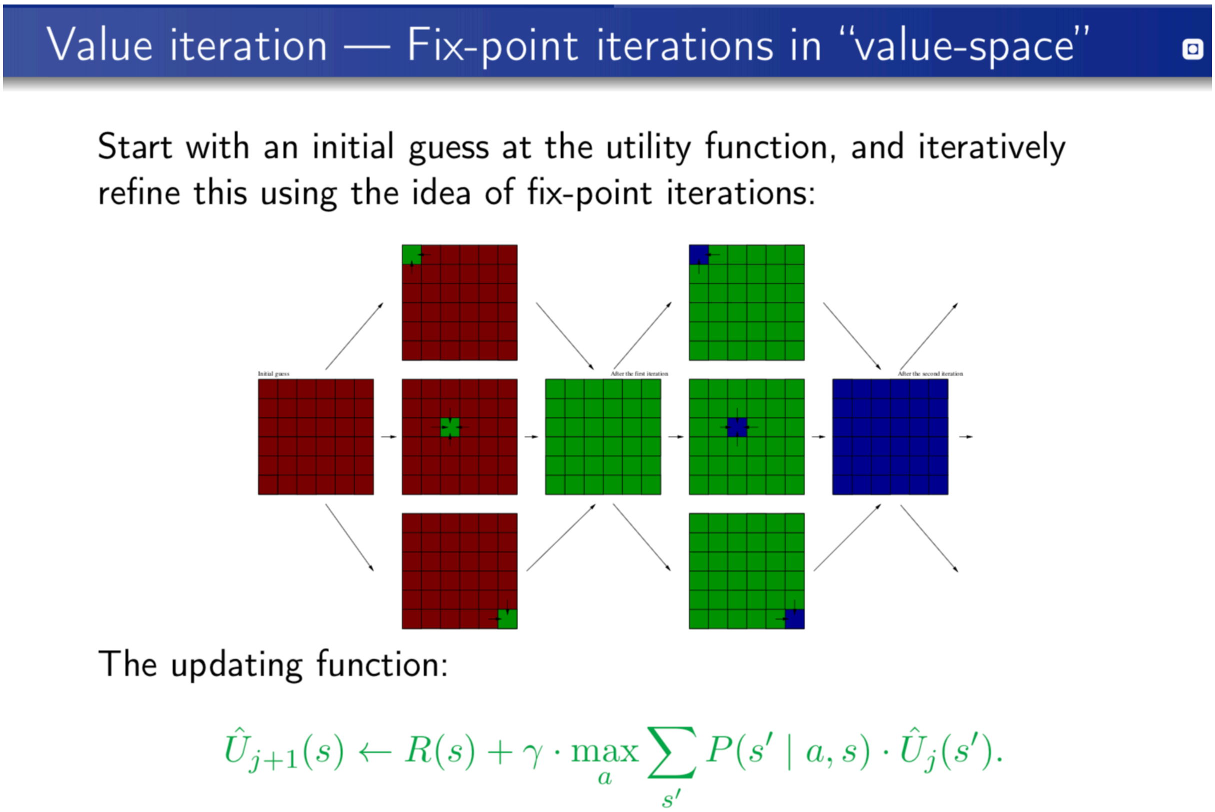
$P(H_2|l_1, l_2) = \alpha P(l_2|H_2)P(H_2|l_1) = \alpha < 0.7{,}0.3 ><0.58{,}0.42> = \alpha < 0.41{,}0.13 >=<0.76{,}0.24>$

d) **Mixing time (2p):** What is mixing time?

Mixing time is the time a temporal model, such as an HMM, uses to reach or get close to the stationary distribution of the Markov process.

Task 4: Making complex decisions

a) **Algorithm (5p):** Describe the value iteration algorithm.



b) **Example (10p):** Execute the value iteration algorithm (two iterations) on the following example where a robot is to navigate on top of a mountain to send signals to its friends in the outer space (10), but with a steep cliff (-7) and some muddy ground (-2):

	1	2	3
1			
2		10	-7
3			-2

The robot uses battery for each grid cell it moves at a cost of -0.2. The robot that can move north, south, east and west with a probability of 0.8 going in the wanted direction and 0.2 probability of going sideways (0.1 for each direction). If it moves into a wall it effectively stands still.

The answer should be twofold: 1) a grid with the resulting utilities and 2) the optimal policy.

Initial guess:

	1	2	3
1	0	0	0
2	0	0	0
3	0	0	0

First iteration:

	1	2	3
1	-0.2	-0.2	-0.2
2	-0.2	10	-7
3	-0.2	-0.2	-2

Second iteration:

The updating function:

$$\hat{U}_{j+1}(s) \leftarrow R(s) + \gamma \cdot \max_a \sum_{s'} P(s' \mid a, s) \cdot \hat{U}_j(s').$$

No learning rate has been specified. **Assumption:** learning rate = 0.9

Sequence: N, S, E, W

U({1,1}) = -0.2 + 0.9 * max(
0.8*-0.2 + 0.1*-0.2 + 0.1*-0.2,

0.8*-0.2 + 0.1*-0.2 + 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2 + 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2 + 0.1*-0.2) = -0.2 + 0.9* max(-0.2, -0.2, -0.2, -0.2) = -0.38

U({1,2}) = -0.2 + 0.9 * max(
0.8*-0.2 + 0.1*-0.2 + 0.1*-0.2,
0.8*10 + 0.1*-0.2 + 0.1*-0.2,
0.8*-0.2 + 0.1*10 + 0.1*-0.2,
0.8*-0.2 + 0.1*10 + 0.1*-0.2) = -0.2 + 0.9* max(-0.2, 7.96, 0.82, 0.82) = 6.96

U({1,3}) = -0.2 + 0.9 max(
0.8*-0.2 + 0.1*-0.2 + 0.1*-7,
0.8*0.2 + 0.1*-0.2 + 0.1*-7,
0.8*-7 + 0.1*-0.2+ 0.1*-0.2,
0.8*-0.2 + 0.1*-7 + 0.1*-0.2) = -0.2 + 0.9* max(-0.2, -5.64, -0.88, -0.88) = -0.38

U({2,1}) = -0.2 + 0.9 max(
0.8*-0.2 + 0.1*-0.2 + 0.1*10,
0.8*-0.2 + 0.1*-0.2 + 0.1*10,
0.8*10 + 0.1*-7+ 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2 + 0.1*0.2) = -0.2 + 0.9* max(0.82, 0.82, 7.28, -0.38) = 6.55

U({2,2}) = 10 + 0.9 max(
0.8*-0.2 + 0.1*-0.2 + 0.1*10,
0.8*-0.2 + 0.1*-0.2 + 0.1*10,
0.8*10 + 0.1*-7+ 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2 + 0.1*0.2) = 10 + 0.9* max(0.82, 0.82, 7.28, -0.38) = 16.55

U({2,3}) = -7 + 0.9 max(
0.8*-0.2 + 0.1*10 + 0.1*-7,
0.8*-2 + 0.1*10 + 0.1*-7,
0.8*-7 + 0.1*-2+ 0.1*-0.2,
0.8*10 + 0.1*-2 + 0.1*0.2) = -7 + 0.9* max(0.14, -1.3, 5.82, 7,82) = 0.038

U({3,1}) = -0.2 + 0.9 max(
0.8*-0.2 + 0.1*-0.2+ 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2+ 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2+ 0.1*-0.2,
0.8*-0.2 + 0.1*-0.2+ 0.1*-0.2,) = -0.2 + 0.9* max(-0.2, -0.2, -0.2, -0.2) = -0.38

U({3,2}) = -0.2 + 0.9 max(
0.8*10 + 0.1*-2+ 0.1*-0.2,
0.8*-0.2 + 0.1*-2+ 0.1*-0.2,

0.8*-2 + 0.1*-0.2+ 0.1*10,
0.8*-0.2 + 0.1*10+ 0.1*-0.2) = -0.2 + 0.9* max(7.78, -0.38, -0.62, -0.82) = 6.80

U({3,3}) = -2 + 0.9 max(
0.8*-7 + 0.1*-2+ 0.1*-0.2,
0.8*-2 + 0.1*-2+ 0.1*-0.2,
0.8*-2 + 0.1*-2+ 0.1*-7,
0.8*-0.2 + 0.1*-7+ 0.1*-2) = -0.2 + 0.9* max(-5.82, -1.82, -2.5, -1.06) = -2.95

Resulting utilities and optimal policy:

Second iteration

	1	2	3
1	-0.38	6.96	-0.38
2	6.55	16.55	0.038
3	-0.38	6.8	-2.95

Optimal policy

	1	2	3
1	→	↓	←
2	→	X	←
3	→	↑	←

Task 5: Decision Trees (25p)

You are hired by the local power company to predict the power production of their wind farm. You are given the data set listed in table 1.

Table 1: Wind power prediction training data set from the local power company.

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
1	H	N	H	C	H
2	L	N	H	C	L
3	M	S	H	S	H
4	L	S	L	S	L
5	M	S	L	C	L
6	H	N	H	C	H
7	M	N	H	S	H
8	M	N	H	C	H
9	H	S	L	S	H
10	M	S	H	S	L

The attributes include the hour of the day the sample data describes (*ID*), the wind farm's production for the previous hour (*prod_prev_hour*), the current wind direction at a measurement location in the farm (*wind_direction*), the forecasted wind strength the next hour (*pred_wind_strength*), the forcasted weather (*weather*) and the actual production for a given hour (*prod*).

You decide that you can use decision trees to predict the power production of the wind farm. Good decision!

a) **Information gain (5p):** Explain information gain and how it can be used to select attributes for a decision tree. Give an example by using one of the attributes in the wind power prediction data set.

Solution:
This is chapter 18.3 in the book, and 18.3.4 for attribute testing.

Information gain is the expected reduction in entropy one gets when a data set is split on an attribute. Constructing a decision tree is all about finding attributes that returns the highest information gain.

Compute the importance of an attribute, A, through information gain:

Importance(attribute) = Gain(A) = B(p/(p+n)) - Remainder(A)

where *p* = positive examples and *n* is negative examples. B(q) is the entropy of a Boolean random variable that is true with probability q:

B(q) = entropy(q) - (q*$\log_2$(q) + (1-q)* $\log_2$(1-q))

and Remainder is:

$$Remainder(A) = \sum_{k=1}^{d} \frac{p_k + n_k}{p + n} B(\frac{p_k}{p_k + n_k})$$

where p_k is the positive and n_k is the negative examples in subset k, where there are k distinct values of attribute A in the training set.

The attribute *prod* is our classification attribute or decision value

p = prod(H) = 6
n = prod(L) = 4

For the attribute prod_prev_hour we have:

IMPORTANCE(*prod_prev_hour)* = B(6/(6+4)) - 3/10*B(0/3) + 2/10*B(2/2) + 5/10*B(2/5)=0.49

b) **Decision tree (12p):** Use information gain to generate a decision based on the provided wind power prediction data set. The answer should be a drawing of the decision tree.

Solution:

Importance calculations for the rest of the attributes:

IMPORTANCE(*wind_direction)* = B(6/(6+4)) - 5/10*B(1/5) + 5/10*B(3/5) = 0.12

IMPORTANCE(*pred_wind_strength)* = B(6/(6+4)) - 6/10*B(1/6) + 4/10*B(3/4) = 0.28

IMPORTANCE(*weather)* = B(6/(6+4)) - 5/10*B(2/5) + 5/10*B(2/5) = 0.00

Choose *prod_prev_hour* as root node. All nodes prod_prev_hour(H) = H and prod_prev_hour(L) = L.

Hence the remaining examples are the rows in white (ID: 3, 5, 7, 8, 10):

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
1	H	N	H	C	H
2	L	N	H	C	L
3	M	S	H	S	H
4	L	S	L	S	L
5	M	S	L	C	L
6	H	N	H	C	H
7	M	N	H	S	H
8	M	N	H	C	H
9	H	S	L	S	H
10	M	S	H	S	L

Next branch:

IMPORTANCE(*wind_direction)* = B(3/(3+2)) - 3/5*B(1/3) + 2/5*B(2/2) = 0.97-0.55-0.0=0.70

IMPORTANCE(*pred_wind_strength)* = B(3/(3+2)) - 3/5*B(3/3) + 2/5*B(1/2) = 0.97-0.28=0.65

IMPORTANCE(*weather)* = B(3/(3+2)) - 3/5*B(2/3) + 2/5*B(1/2) = 0.97-0.24-0.28=0.50

Choose *wind_direction* as next branching node.

The remaining examples are the rows in white (ID: 3, 5, 10):

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
1	H	N	H	C	H

2	L	N	H	C	L
3	M	S	H	S	H
4	L	S	L	S	L
5	M	S	L	C	L
6	H	N	H	C	H
7	M	N	H	S	H
8	M	N	H	C	H
9	H	S	L	S	H
10	M	S	H	S	L

Next branch:

IMPORTANCE(*pred_wind_strength)* = B(2/(3)) - 2/3*B(1/2) + 1/3*B(1/1) = 0.92-0.67-0.0=0.25

IMPORTANCE(*weather)* = B(2/(3)) - 2/3*B(1/2) + 1/3*B(1/1) = 0.92-0.67-0.0=0.25

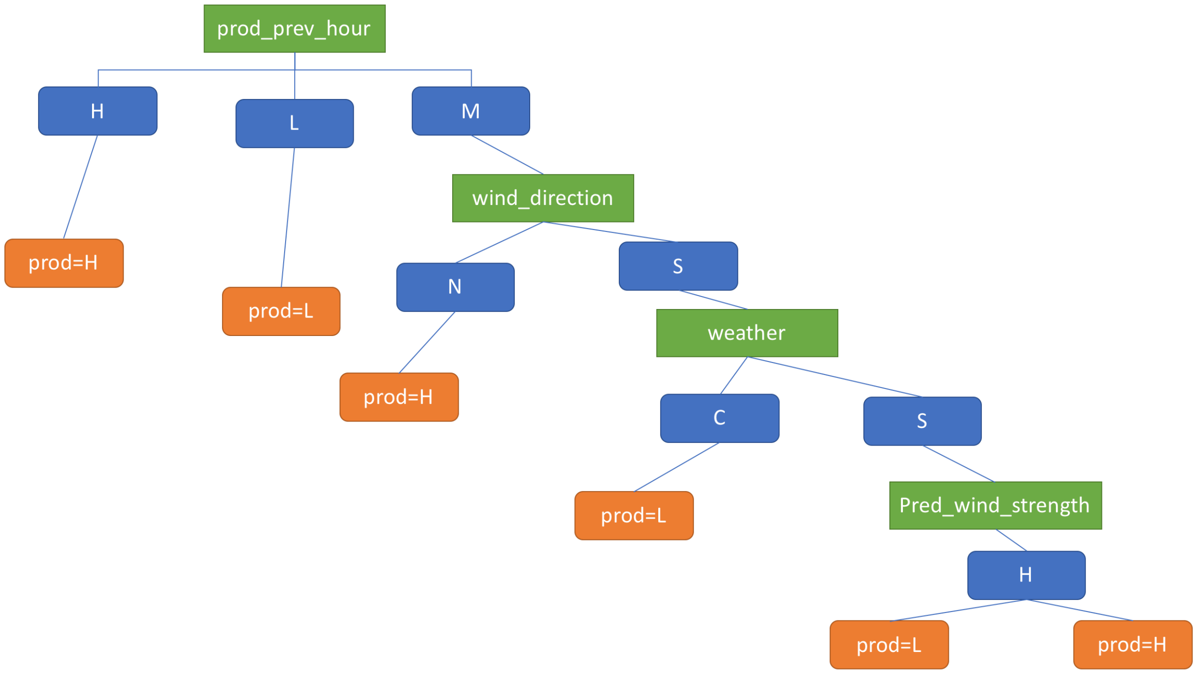
Choose either one of *weather* and *wind_direction* for the last one, as both have information gain = 0.25. We choose *weather*.

The remaining examples are the rows in white (ID: 3, 10):

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
1	H	N	H	C	H
2	L	N	H	C	L
3	M	S	H	S	H
4	L	S	L	S	L
5	M	S	L	C	L
6	H	N	H	C	H
7	M	N	H	S	H
8	M	N	H	C	H
9	H	S	L	S	H
10	M	S	H	S	L

Rows 3 and 10 cannot be classified. Classification will be either H or L. Choose randomly with probability of 0.5 for each class.

Resulting decision tree:



c) **Evaluation (3p):** The local power company wants to test your prediction algorithm and provides some test data listed in table 2.

Table 2: Wind power prediction test set.

ID	prod_prev_hour	wind_direction	pred_wind_strength	weather	prod
11	M	S	H	S	M

Run your decision tree on the data. Show the executing path in your decision tree. What is your prediction?

Solution:
prod_prev_hour = M => wind_direction = S => weather = S => prod_wind_strength = H => class = H OR L with 0.5 probability.

Prediction is either H or L, and it should be M.

d) **Advice (5p):** The local power company is not happy with your result, but you explain to them that the experiment has some flaws. Please advise them on how to improve the experiment.

Solution:

More data: Too little data to cover all possible situations. Examples with test class not in training set.

__Task 6: Reinforcement learning__

a) **Q-learning algorithm (5p):** Give a short description of the Q-learning algorithm and provide pseudo code.

Q-learning algorithm – deterministic world

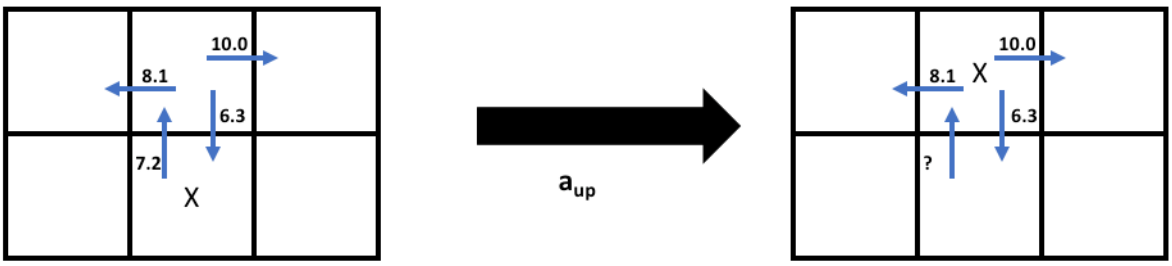
1. For each s, a initialize table entry $\hat{Q}(a, s) = 0$
2. Observe current state s
3. Do forever:
   - Select an action a and execute it
   - Receive immediate reward r
   - Observe the new state $s' := \delta(a, s)$
   - Update the table entry for $\hat{Q}(a, s)$ as follows:

$$\hat{Q}(a, s) := R(s) + \gamma \cdot \max_{a'} \hat{Q}(a', s')$$

   - $s := s'$

b) **Example (5p):** An agent X is located in state s1 (bottom, middle) is doing Q-learning with fuel expenditure of 0.1 and a discount factor of 0.9. X performs an action a_{up} and ends up in s_2 (top, middle):

Use this information to update $Q(a_{up}, s_1)$.



Solution:

$$\hat{Q}(a_{up}, s_1) = \mathrm{R}(s_1) + \gamma \cdot max_{a'}\, \hat{Q}\,(a', s_2) = -0.1 + 0.9 \cdot max\{6.3, 8.1, 10.0\} \ = \ 8.9$$