Lunds Tekniska Högskola
Institutionen för datavetenskap

EDAP01 (a.k.a. TFRP20)
VT 2023

Artificiell Intelligens
Artificial Intelligence
Tentamen 2023–03–15, 14.00–19.00, MA10 I + J

You can give your answers in English or Swedish.
You are welcome to use a combination of figures and text in your answers.

1 Machine Learning (PN) 30 points

1.1 Notations

In this exercise, you will consider a dataset of N observations, where your will try to predict an observed value (output or response) y_i from an input vector $\mathbf{x}_i$. The dataset, denoted X, is the matrix of the input vectors arranged by rows and $\mathbf{y} = (y_1, y_2, ..., y_N)$ is the vector of observed values.

Linear regression is a linear model that predicts y_i using a dot product between $\mathbf{x}_i$ and a learnable weight vector $\mathbf{w}$. We denote $\hat{y}_i$ this predicted value and we have then $\mathbf{x}_i \cdot \mathbf{w} = \hat{y}_i$. We define the error as the predicted value minus the observed value: $\hat{y}_i - y_i$.

1.2 Background

The objective of linear regression is to find the $\mathbf{w}$ vector that minimizes the quadratic loss defined as the sum of the squared errors, L_2. During the lectures, we have seen that this minimization has a closed-form for solution with the pseudoinverse:

$$\mathbf{w} = (X^\intercal X)^{-1} X^\intercal \mathbf{y}.$$

In the loss formulation of L_2, as well as in the logistic loss, the norm of the weight vector, $||\mathbf{w}||$, is not bounded. In many real cases, when columns of the dataset are close to be linearly dependent, the weight coordinates can take huge values when computing a pseudoinverse. This is also the case when using a gradient descent depending on the initial conditions.

During the lectures, we saw one way to mitigate this problem by adding an identity matrix scaled by a small λ value:

$$\mathbf{w} = (X^\intercal X + \lambda I)^{-1} X^\intercal \mathbf{y}$$

In your assignment program, instead of a pseudoinverse, you used gradient descent to find $\mathbf{w}$. In this examination, you will adapt regularization to a gradient descent and derive a new update rule.

1.3 Linear regression

For linear regression, the L_2 loss, corresponding to the sum of squared errors, corresponds to:

$$\begin{array}{rcl} Loss(\mathbf{w}) & = & \sum_i (\hat{y}_i - y_i)^2, \\ & = & \sum_i (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2. \end{array}$$

To damp the norm of the weights during the descent, we can reformulate the norm as a cost that incorporates a fraction of the norm value:

$$\begin{array}{rcl} Cost(\mathbf{w}) & = & \sum_i (\hat{y}_i - y_i)^2 + \lambda ||\mathbf{w}||^2, \\ & = & \sum_i (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2 + \lambda ||\mathbf{w}||^2. \end{array}$$

This process is called *regularization.*

1.4 Gradient descent with regularization

In this exercise, you will rewrite the update rule of a gradient descent to incorporate regularization. You will limit yourself to a two-dimensional space and straight lines where:

$$\hat{y}_i = w_0 + w_1 x_i.$$

1.4.1 Partial derivatives

1. Compared with the loss, explain why the cost definition will limit the range of the weight parameters; 2 points
2. Using y_i and $\hat{y}_i$, write the L_2 loss for one point; 1 point
3. Using y_i, x_i, and $\mathbf{w} = (w_0, w_1)$, rewrite this L_2 loss for one point; 1 point
4. Compute the partial derivatives of the L_2 loss with respect to w_0 and w_1, $\dfrac{\partial Loss}{\partial w_0}$ and $\dfrac{\partial Loss}{\partial w_1}$, for one point; 2 points
5. Using y_i, x_i, $\mathbf{w} = (w_0, w_1)$, and λ, write the cost for one point; 1 point
6. Compute the partial derivatives of the cost with respect to w_0 and w_1, $\dfrac{\partial Cost}{\partial w_0}$ and $\dfrac{\partial Cost}{\partial w_1}$, for one point. 3 points

1.4.2 Gradient descent

The gradient descent update rule at step t is given by:

$$\mathbf{w}_{(t+1)} = \mathbf{w}_{(t)} - \alpha \nabla_{\mathbf{w}} Loss(\mathbf{w}_{(t)})$$

1. Give the update rules for w_0 and w_1 with the loss 1 point

$$\begin{array}{rcl} w_0 & \leftarrow & ? \\ w_1 & \leftarrow & ?; \end{array}$$

2. Give the update rules for w_0 and w_1 with the cost: 3 points

$$\begin{aligned} w_0 &\leftarrow ? \\ w_1 &\leftarrow ?; \end{aligned}$$

1.5 Optimal model

We saw during the lectures and the assignment that we reach an optimal value of the weight vector $\mathbf{w}$ when the gradient is 0[1].

1. Using the results in Sect. 1.4.1, rewrite the equation system

   $$\nabla_{\mathbf{w}} Loss(\mathbf{w}_{(t)}) = 0$$

   with x_i, y_i, w_0 and w_1; 2 points

2. What does this new equation system mean in terms of loss? 2 points

3. Rewrite the equation system

   $$\nabla_{\mathbf{w}} Cost(\mathbf{w}_{(t)}) = 0$$

   with x_i, y_i, λ, w_0 and w_1; 3 points

1.6 Relation with the pseudoinverse

The definition of the pseudoinverse with regularization we saw during the lectures is:

$$\mathbf{w} = (X^\intercal X + \lambda I)^{-1} X^\intercal \mathbf{y}$$

1. Expand the equation below

   $$(X^\intercal X + \lambda I)\mathbf{w} = X^\intercal \mathbf{y}$$

   in a two-dimensional space with one point, x_i, i.e. X is limited to one point $(1, x_i)$. 4 points

   Note that the matrix product of a column vector by a row vector is:

   $$\begin{bmatrix} a \\ b \end{bmatrix} \begin{bmatrix} c & d \end{bmatrix} = \begin{bmatrix} ac & ad \\ bc & bd \end{bmatrix}$$

2. Show it is equivalent to what you obtained in the system

   $$\nabla_{\mathbf{w}} Cost(\mathbf{w}_{(t)}) = 0.$$

   2 points

[1] In the program, you probably stopped the descent when the gradient norm was below a small ϵ value.

1.7 Logistic regression

We can also regularize logistic regression with the squared norm of the weight vector.

The logistic loss for one observation is defined as:

$$-y_i \ln \hat{y}_i - (1 - y_i) \ln(1 - \hat{y}_i),$$

where

$$\hat{y}_i = \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}_i}}.$$

We regularize this loss with the addition of the squared norm of the weight vector. For one observation, this would yield:

$$-y_i \ln \hat{y}_i - (1 - y_i) \ln(1 - \hat{y}_i) + \lambda ||\mathbf{w}||^2.$$

1. Tell how you would modify the update rule of logistic regression to have a regularized gradient descent. 3 points

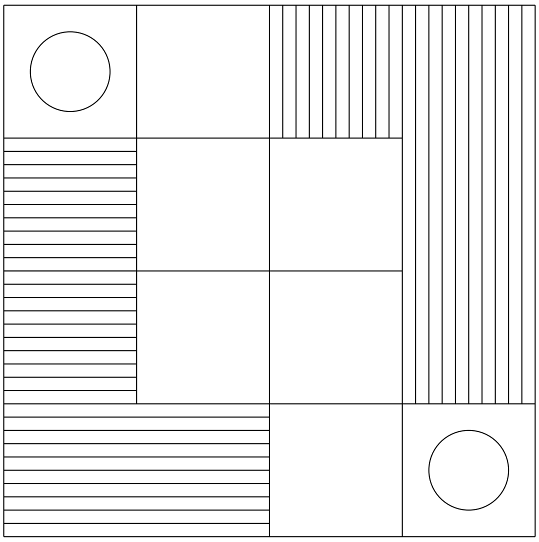
Note: You do not need to compute or write the gradient, just write how you would modify the existing update rule.

2 Games (JM) 15p

L-game is played by two players on a 4 × 4 board. Each player has an L-shaped figure which can be turned upside down and/or rotated in all directions. A move consists of two parts:

1. The player lifts her L-figure and puts it down on the board in a different position than before.
2. If she wants, she can also move one of the two neutral pieces[2] to a new position.

The player who cannot move her L to a new position, looses the game. The following picture illustrates the board in the beginning of the game.



Task 1 Assume now that you want to write a program that could play the L-game. Your task consists of representing the problem as an *adversarial search* problem. Describe how state and operators (possible moves) could be represented, how the goal state will be recognized (a *goal-test* function), how possible moves will be generated (a *successor* function), how one of the possible moves will be chosen (a *choose-move* function), etc. 10 points

In order to avoid misunderstanding, some precision is necessary in your answer. Therefore it would be a benefit if you could use e.g. list structures (or whatever you deem appropriate) to define the necessary data types you choose to represent state and operators. You can write your functions using a pseudocode.

Remember that your program is going to play against an opponent. Therefore during the search for the next best move you should take into account the possible moves of the opponent.

[2] A neutral piece, marked by a circle on top of it, covers just one square of the board.

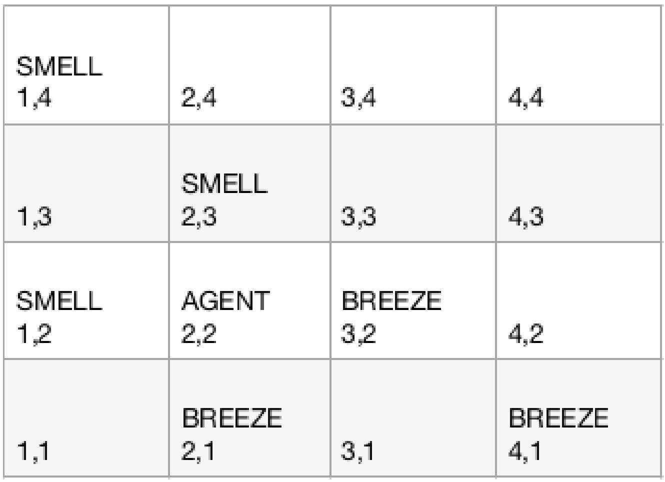
Task 2 What is the branching factor of the search space? Is the search space finite? 2 points

How would your answers to these two question change if the neutral pieces were not moved by any of the players, but got random placement after each player's move of her L-piece? I.e., the game would go as follows: (1) first player moves her L-piece, (2) the two neutral pieces get random positions on the available space (including their current placement), (3) the other player moves her L-piece, (4) the two neutral pieces get random positions on the available space, (5) do (1) again. 2 points

Task 3 Can you come up with a winning strategy for the player moving first? 1 point

3 Logic (JM) 20p

Imagine the following situation in the wumpus world:



SMELL 1,4	2,4	3,4	4,4
1,3	SMELL 2,3	3,3	4,3
SMELL 1,2	AGENT 2,2	BREEZE 3,2	4,2
1,1	BREEZE 2,1	3,1	BREEZE 4,1

Prove that the position 3,3 is safe, i.e., the agent will not get killed if it moves there (via position 3,2 or position 2,3; other paths may be unsafe, as you know). You need to

- formulate your problem in logic, 4 points
- state all *necessary* laws of the Wumpus world (do not do more than necessary, you don't have time), 6 points
- and finally prove that position 3,3 is safe. Maximal number of points (10) will be given for a resolution proof. 10 points

4 KR (JM) 5 p

How would the problem above be represented in the semantic web setting? Try to give a concrete example rather than just describe the idea. Would the same pattern of reasoning as in previous question be possible in this case? 5 points

5 Probabilistic reasoning, BNs (EAT), 9p

Note: The figures referred to in the question (networks i) – iv)) may be found on pages 9 and 10.

You have a set of 5 random variables. You know the following about them:

- Semantically speaking, the phenomenon represented by B is known to cause an effect on what is represented by C, which can then have an effect on what is represented by D and A respectively.
- $P(D|A, B, C, E) = P(D|B)$ and $P(A|B, C, D, E) = P(A|B)$
- $P(E|A, B, C, D) = P(E)$
- $P(D, A) \neq P(D)P(A)$

Answer the following questions (motivate your answers!):

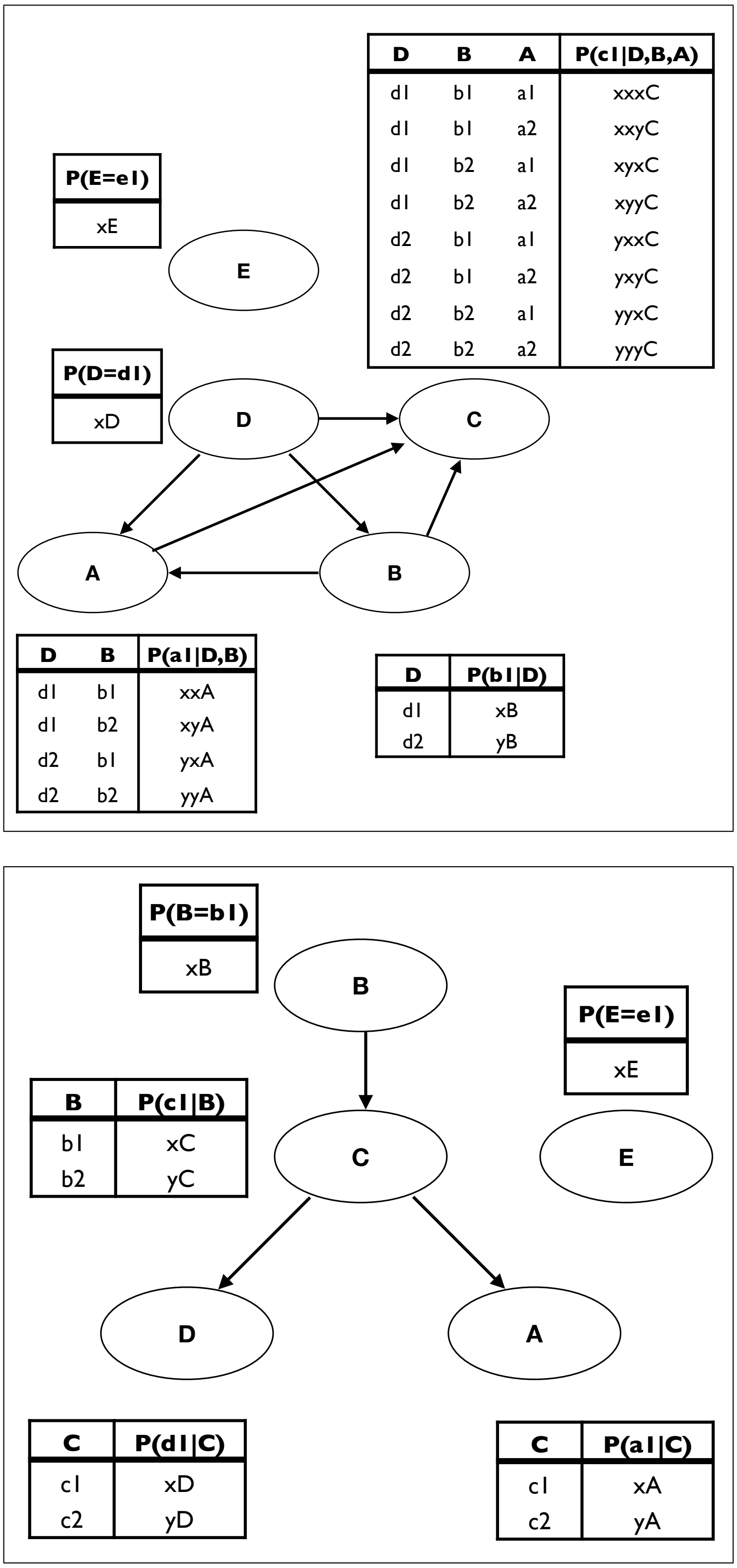
a) When are two random variables independent of each other? When are they conditionally independent? 2 points

b) Which of the networks i), ii), and iii) is / are correct wrt the set of variables described above? 2 points

c) Which network is optimal (if any), and why? 2 points

d) What do the CPTs represent? Explain explicitly the one for variable C in network i)! 2 points

e) Network iv) represents a special case of a Bayesian Network. What makes it special and how is this type of network also called? 1 point



P(E=e1)
xE

D	B	A	P(c1|D,B,A)
d1	b1	a1	xxxC
d1	b1	a2	xxyC
d1	b2	a1	xyxC
d1	b2	a2	xyyC
d2	b1	a1	yxxC
d2	b1	a2	yxyC
d2	b2	a1	yyxC
d2	b2	a2	yyyC

P(D=d1)
xD

D	B	P(a1|D,B)
d1	b1	xxA
d1	b2	xyA
d2	b1	yxA
d2	b2	yyA

D	P(b1|D)
d1	xB
d2	yB

P(B=b1)
xB

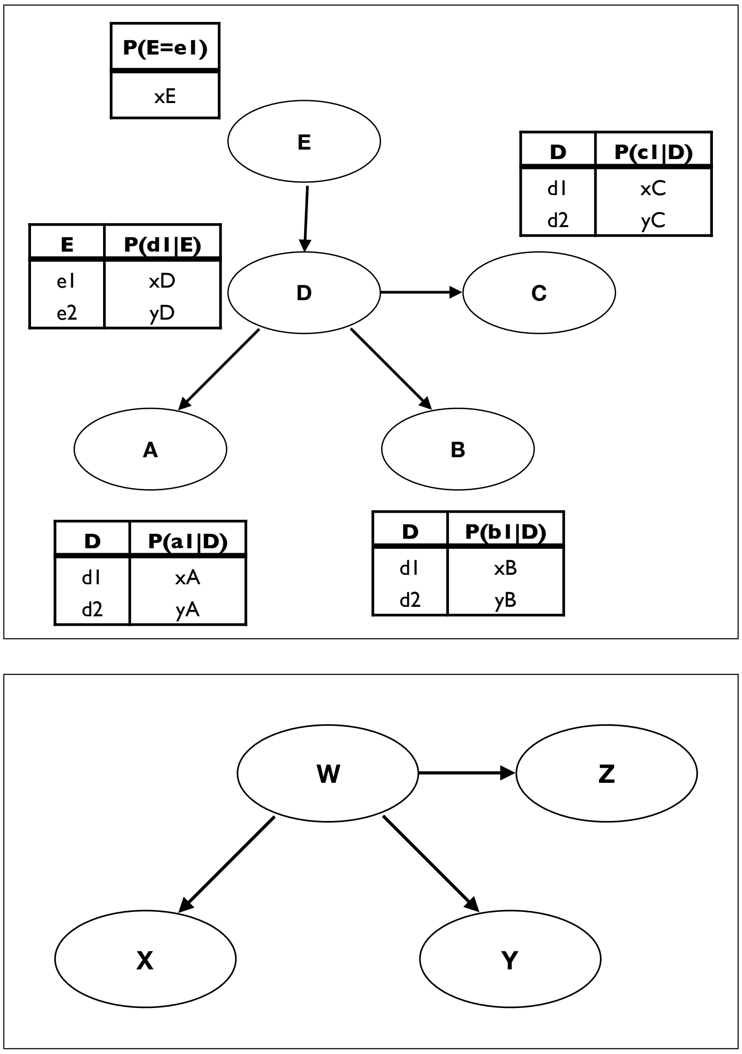
P(E=e1)
xE

B	P(c1|B)
b1	xC
b2	yC

C	P(d1|C)
c1	xD
c2	yD

C	P(a1|C)
c1	xA
c2	yA

Figure 1: Networks i) (top), ii) (bottom)



P(E=e1)
xE

E	P(d1|E)
e1	xD
e2	yD

D	P(c1|D)
d1	xC
d2	yC

D	P(a1|D)
d1	xA
d2	yA

D	P(b1|D)
d1	xB
d2	yB

Figure 2: Networks iii) (top) and iv) (bottom

6 Probabilistic (Bayesian) reasoning (over time) / Robotics (EAT), 21p

Similar to what you did before as a homework assignment, you are supposed to localise an agent in a grid world. In this case, the world has obstacles in it, and the states correspond to the number of possible positions for the agent (see below). The agent moves according to the following assumptions: Stay put with probability 0.4, and move to one directly adjacent state (no diagonal moves) with probability 0.6 overall. It reports in every step how many spots it could move to around it (this means, it reports "1", "2", or "3") with probability 0.8, or it tries to trick you and does not report anything with probability 0.2.

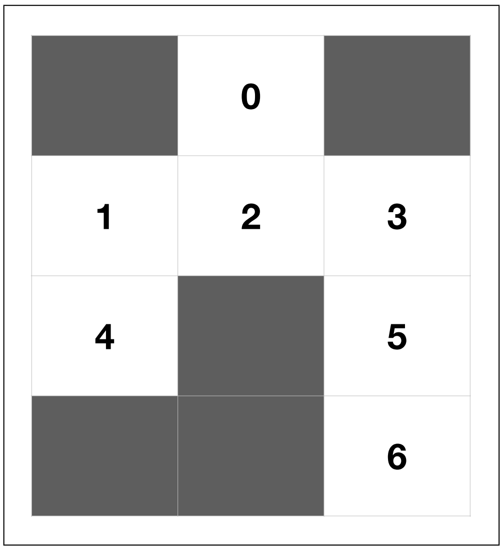


Figure 3: The gridworld for the agent to move in; the numbers in the cells correspond to the state

a) Formulate the problem of localising the agent as an HMM in matrix-vector form, i.e. note down the matrices describing the transition and sensor models. 6 points

b) Assume you start out with not knowing where the agent actually is, but then you get a sensor report of "3". Do you know for sure now, where it is? Why? Explain both intuitively and mathematically! 2 points

c) Assume now to receive the following series of sensor reports AFTER the initial report of "3" discussed above: "2", "nothing", "1". What is/are the *possible* state(s) the agent can be in? Explain intuitively! 2 points

d) Using your models, determine and explain the *most likely* state for the agent after the report series above ("2", "nothing", "1"), assuming again to have gotten report "3" initially. 8 points

e) What is the Markov assumption? What does this mean for the *transition model* of a process? What does it mean for the *sensor model*? 1 point

f) How does *forward filtering* work and what can it be used for? 2 points

Good Luck!