Trial Exam IN3050/4050 Spring 2021

Hi IN3050/4050-students! Below, we have made a trial exam consisting of questions representative for what you will see in this year's exam. Many of them have been used before (e.g. in last year's exam/trial exam), but these are all questions that we believe would also be fitting for this year's exam, and you can expect to see a similar style of problems on this year's final exam.

A difference between this and the final exam, however, is that the final exam will be given and delivered in Inspera. So, please take some time to familiarize yourself with how to do exams in Inspera if you have not done this already. Some links to useful information:

- Examination guidelines for Spring 2021 at the Faculty of Mathematics and Natural Sciences
- UiO's page for preparing for exams in Inspera
- Each question in our exam will be answered in the "long form assignment" style demonstrated here.
- We encourage you to log in to Inspera and test their demo exams, especially the "long form assignment" demo. Accessible here after logging in. In particular, have a look at the equation editor, with which it may be useful to have a bit of practice well before the exam.

Simulated Annealing (6p)

In simulated annealing,

a) What would happen if we start with a very low temperature (keeping low through search)? Which search algorithm would this be like? (3p)

b) What would happen if we start with a very high temperature and never decrease? (3p)

Master Students Only: Search (5p)

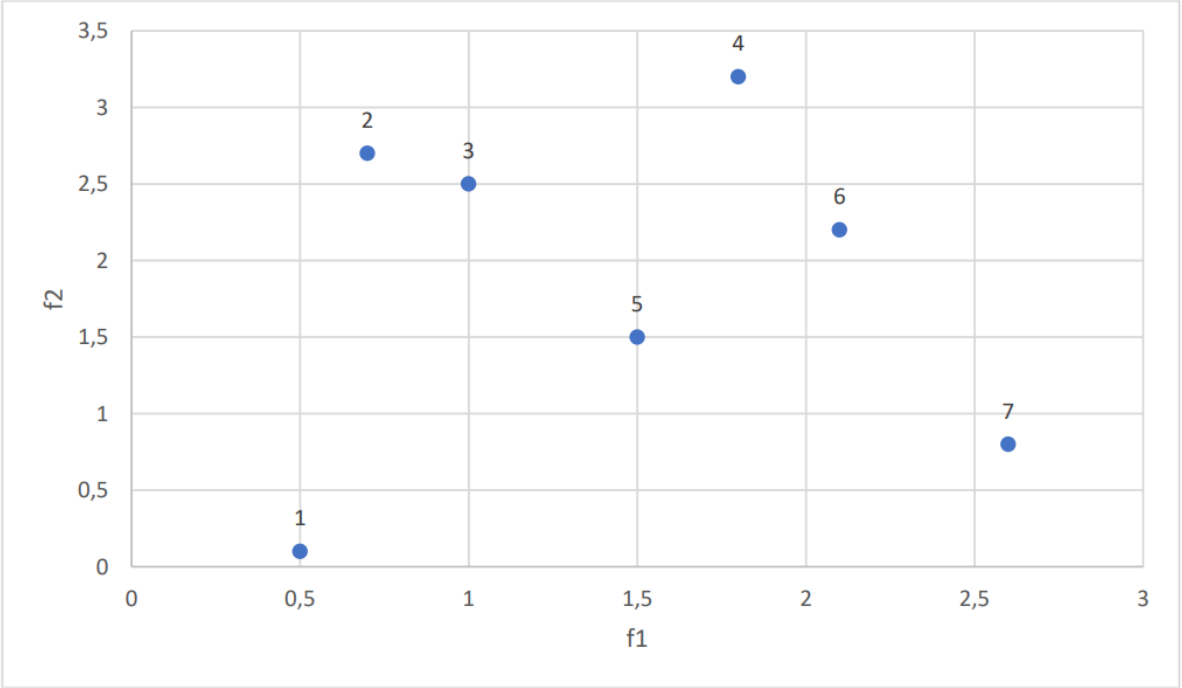
In a few sentences, sketch how you could modify a hill climbing algorithm in order to improve chances of finding the global optimum.

Bachelor Students only: EA Selection (5p)

Five strings have the following fitness values: 3, 6, 9, 12, 15. Under Fitness Proportionate Selection, compute the expected number of copies of each string in the mating pool if a constant population size, n = 5, is maintained.

Pareto Optimality (9p)

For an optimization problem we wish to optimize solutions according to two different objectives, f1 and f2. The fitness values according to the two objectives for 7 different solutions are plotted in the figure below.



a) What requirements do the solutions in a Pareto optimal set need to fulfill? (3p)

Find the Pareto optimal set of solutions when

b) Maximizing f1 and f2 (3p)
c) Maximizing f1 but minimizing f2 (3p)

Perceptron and linear regression classifier (12p)

Given the following data

Item	x1	x2	Class
A	1	2	yes =1
B	2	1	yes =1
C	1	1	no =0
D	1	0	no =0

a) Are the data linearly separable? State reasons for your answer. (4p)

b) We will train a perceptron on the data. We add a bias $x_0 = -1$ to each of the data points. Suppose the current weigths to be $\boldsymbol{w} = (0, -1, 1)$. Assume a learning rate of 0.1. How should the weights be updated if point A is considered? How would the weights have been updated if the algorithm instead had considered point B? (4p)

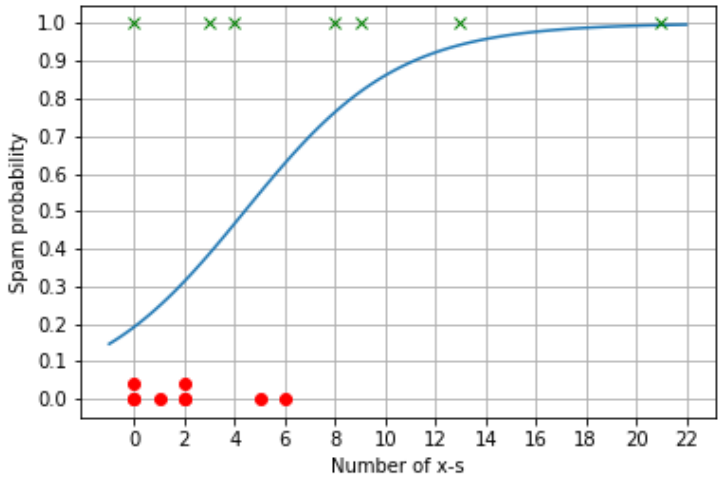
c) Say, we instead had applied a linear regression classifier. How should the weights have been updated when considering datapoint A, again assuming a learning rate of 0.1. And how would they have been updated if we instead considered point B? (4p)

Logistic Regression (12 p)

Kim is building a spam filter. She has the hypothesis that counting the occurrences of the letter 'x' in the e-mails will be a good indicator of spam or no-spam. She collects 7 spam messages and 7 no-spam messages and counts the number of x-s in each. Here is what she finds.

- Number of 'x'-s in each spam: [0, 3, 4, 8, 9, 13, 21]
- Number of 'x'-s in each no-spam: [0, 0, 1, 2, 2, 5, 6]

She trains a logistic regression classifier on the data and plots the classifier against the data.



Assume the logistic regression model and answer the following questions:

a) Consider an e-mail with no 'x'-s. According to the model, what is roughly the probability of this message being a spam message and what is the probability of it not being a spam. (3p)

b) How many x-s must an e-mail contain to guarantee it is a spam mail? (3p)

c) How is a logistic regression model normally turned into a binary classifier? If you turn the model into a classifier in this way, what is the accuracy of the classifier on the training data? (3p)

d) It is most important that no no-spams are classified as spams. How can this goal be described in terms of precision and recall? How can the logistic regression classifier be modified to try to achieve this goal? (3p)

MLP and Back-propagation (10p)

a) Figure 1 shows the Multi-layer Perceptron Algorithm as presented by Marsland in the course book. As presented, this is an algorithm for classification. Suppose you instead will use an MLP for regression. Which lines in the algorithm do you have to change? What are their new forms? (5p)

b) Which activation function does Marsland's algorithm apply in the hidden layer? Suppose you instead will use the RELU activation function in the hidden layer. Which lines do you have to change and what will they look like with RELU? (5p)

The Multi-layer Perceptron Algorithm

- **Initialisation**
  - initialise all weights to small (positive and negative) random values
- **Training**
  - repeat:
    - for each input vector:

      Forwards phase:

      - compute the activation of each neuron j in the hidden layer(s) using:

$$h_\zeta = \sum_{i=0}^{L} x_i v_{i\zeta} \tag{4.4}$$

$$a_\zeta = g(h_\zeta) = \frac{1}{1+\exp(-\beta h_\zeta)} \tag{4.5}$$

      - work through the network until you get to the output layer neurons, which have activations (although see also Section 4.2.3):

$$h_\kappa = \sum_{j} a_j w_{j\kappa} \tag{4.6}$$

$$y_\kappa = g(h_\kappa) = \frac{1}{1+\exp(-\beta h_\kappa)} \tag{4.7}$$

      Backwards phase:

      - compute the error at the output using:

$$\delta_o(\kappa) = (y_\kappa - t_\kappa)\, y_\kappa (1 - y_\kappa) \tag{4.8}$$

      - compute the error in the hidden layer(s) using:

$$\delta_h(\zeta) = a_\zeta (1 - a_\zeta) \sum_{k=1}^{N} w_\zeta \delta_o(k) \tag{4.9}$$

      - update the output layer weights using:

$$w_{\zeta\kappa} \leftarrow w_{\zeta\kappa} - \eta \delta_o(\kappa) a_\zeta^{\text{hidden}} \tag{4.10}$$

      - update the hidden layer weights using:

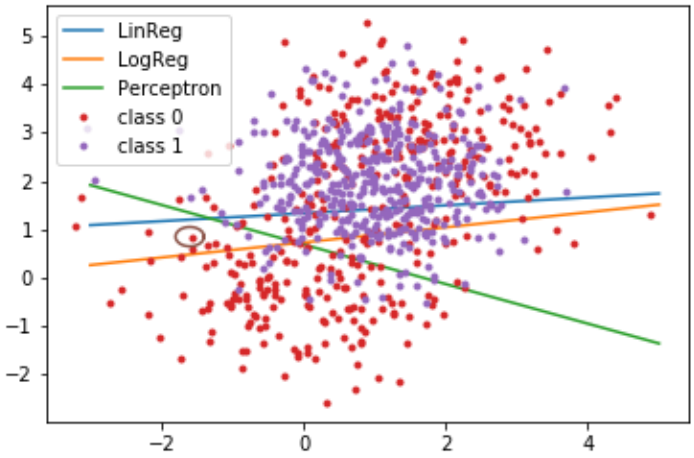
$$v_\iota \leftarrow v_\iota - \eta \delta_h(\kappa) x_\iota \tag{4.11}$$

    - (if using sequential updating) randomise the order of the input vectors so that you don't train in exactly the same order each iteration
  - until learning stops (see Section 4.3.3)
- **Recall**
  - use the Forwards phase in the training section above

Figure 1

Majority Voting Classifier (8 p)

We have trained three different classifiers on the same training data (from mandatory assignment 2); a linear regression classifier, a logistic regression classifier, and a perceptron classifier. We have plotted the decision boundaries for all three classifiers on the training data in the figure. They all classify the points above their boundaries as class 1 (purple) and the points below the boundary as class 0 (red). By referring to the figure and the circled point, explain how a majority voting classifier works



Master's students only: Regularization (8p)

In several forms of supervised machine learning we find a model with a set of weights. The goal of training is to find the weights which best fit the training data $(\boldsymbol{X}, \boldsymbol{t})$. To determin what we mean by best fit, we introduce a loss function, L, and the goal is to determine the weights which minimize the loss, in symbols $\widehat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmax}}(-L(\boldsymbol{X}, \boldsymbol{t}, \boldsymbol{w}))$

In reguarization, one replaces the objective $-L(\boldsymbol{X}, \boldsymbol{t}, \boldsymbol{w})$ with another objective, e.g., for L2-regularization one replaces it with $-L(\boldsymbol{X}, \boldsymbol{t}, \boldsymbol{w}) - \alpha\|\boldsymbol{w}\|^2$.

a) Why does one apply regularization, and what is achieved by regularization?

b) How is α determined?

Bachelor students only: Training and test sets (8p)

Describe what is meant by a training set, test set and development test set in supervised machine learning, and how they are used.

Unsupervised Learning (8p bachelor / 8p master)

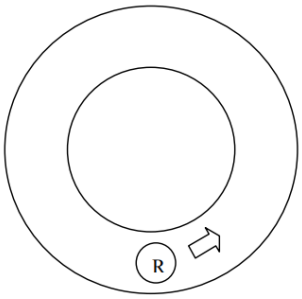
a) **(Master Students Only)** An application of unsupervised learning discussed in class and in the exercises is dimensionality reduction. Sometimes, however, it may be necessary to project data from a lower-dimensional space to a higher-dimensional space. Consider the two algorithms that you have used for dimensionality reduction: PCA and autoencoders; can they be used to generate higher dimensional representations? Briefly explain why/why not. (4p)

b) **(Bachelor Students Only)** Overfitting is a central problem in learning. Suppose you have been given an unlabeled data set containing 1000 samples in 20 dimensions. You ran the K-means algorithms on it using 900 centroids, and you achieved a satisfying matching. Now, someone examines your work, and states that your algorithm is overfitting the data. What does she mean? How is this related to overfitting? (4p)

c) (**All students)** Big data is undoubtedly an important driver for machine learning. However, in certain situations there may be limits on the memory space or the computational power available (think of embedded systems, for instance). In these cases, we may prefer running our algorithms on few selected datapoints. Suppose you have been given an unlabeled data set containing 1000 samples in 20 dimensions, and someone has told you that they want only 10 representative samples (*prototypes*) to run their analysis. Out of the unsupervised algorithms you have studied (PCA, K-means, autoencoders), which one would you use and how? (4p)

Evolutionary Algorithms and Reinforcement Learning (13p)

We would like to set up a neural network (multilayer perceptron) for robot control. The inputs to the neural network are measurements from range sensors, and the output is a direction of movement. The robot is inserted into the circular maze shown to the right, and the goal is to enable it to drive in the direction of the arrow, getting as far as possible within a given time limit, while colliding with the walls as few times as possible.



a) One way to design this neural network is by use of an evolutionary algorithm (EA). The individuals in the population will be possible robot controlling networks that get their fitness computed in simulation. The "lifetime" of one EA individual (a neural network) thus consists of many timesteps, where in each timestep the individual receives inputs from sensors, and calculates outputs (by feeding inputs forward through its connections) that are given as speed settings to the robot's wheels. The wheels then follow this setting until a new output is available. Each individual gets N timesteps to try to drive as far as possible. Assuming that the structure of the network is already specified, briefly describe how you could allow an EA to find the proper weights for this neural network. Include in your description a possible choice for:

a1) the genetic representation (genotype)

a2) variation operators. Include both their names and a brief description of how they work

a3) which measurements to include in the fitness function. You can assume the robot, or the simulator, can gather any physical measurements of relevance to fitness calculation

(6 points, 2 per sub-task)

b) A different way to solve this problem is to apply reinforcement learning (RL). Describe how you would model this problem as a reinforcement learning problem, including how you would define rewards, states, and actions. The RL *algorithm* is not to be described. Note that you should not describe this as a deep reinforcement learning problem, or with other function approximation techniques – rather, formulate it as a discrete RL problem like the examples from the textbook and lectures. (7 points)

Particle Swarm Optimization (PSO) and Developmental Systems (9p)

a) Describe what happens when the position of all particles in PSO are set to the same value of a local optimum (Think about fitness landscapes). How could you adjust PSO to get out of the local optimum to potentially find the global optimum without resetting the particles to random positions initially? Describe your approach in up to 100 words. (3p)

b) An L-System is a parallel rewrite method. Describe how from an alphabet {h,j}, the axiom h will be rewritten when using the rewrite rules: h->jjh and j->h. Write down three iterations/recursions. (3p)

c) When visualizing a string of an L-System, it is useful to implement a bracketed L-System. Describe what '+', '-', '[' and ']' in such L-Systems are used for. (3p)