Department of Mathematics and Computer Science
University of Southern Denmark, Odense

October 12, 2016
Marco Chiarandini

# DM534 - Introduction to Computer Science

## Training Session, Week 41-43, Autumn 2016

---

### Exercise 1. $k$-Nearest Neighbors: Prediction

(Based on slide 21)
Suppose you are trying to predict a response $y$ to an input $x$ and that you are given the set of training data $D = [(x_1, y_1), \ldots, (x_{11}, y_{11})]$ reported and plotted in Figure 1.

$$D = \begin{bmatrix} (8, & 8.31) \\ (14, & 5.56) \\ (0, & 12.1) \\ (6, & 7.94) \\ (3, & 10.09) \\ (2, & 9.89) \\ (4, & 9.52) \\ (7, & 7.77) \\ (8, & 7.51) \\ (11, & 8.0) \\ (8, & 10.59) \end{bmatrix}$$

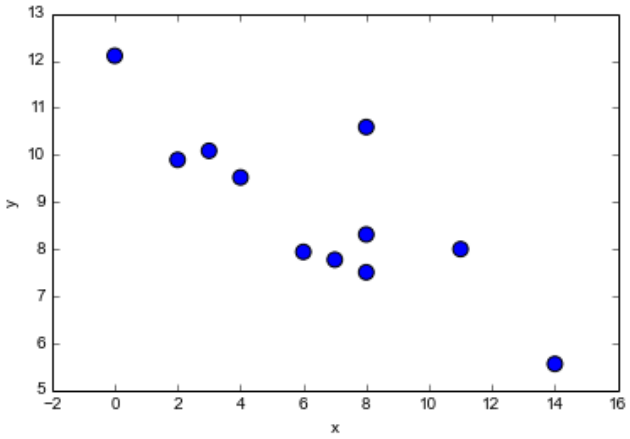


Figure 1: The data for Exercise 1.

Using 5-nearest neighbors, what would be the prediction on an new input $\hat{x} = 8$?
What form of learning is this exercise about?

- Supervised learning, regression
- Supervised learning, classification
- Unsupervised learning
- Reinforcement learning

### Exercise 2. $k$-Nearest Neighbors: Prediction

(Based on slide 21)
Suppose you are trying to predict the class $y \in \{0, 1\}$ of an input $(x_1, x_2)$ and that you are given the set of training data $D = [((x_1, x_2), y_1), \ldots, ((x_{11,1}, x_{11,2}), y_{11})]$ reported and plotted in Figure 2.
Using the 5-nearest neighbors method, what would be the prediction on the new input $\hat{\vec{x}} = (5, 10)$?
What form of learning is this exercise about?

- Supervised learning, regression
- Supervised learning, classification
- Unsupervised learning
- Reinforcement learning

$$D = \begin{bmatrix} ((10, & 2), & 1) \\ ((15, & 2), & 1) \\ ((6, & 11), & 1) \\ ((2, & 3), & 0) \\ ((5, & 15), & 1) \\ ((5, & 14), & 1) \\ ((10, & 1), & 0) \\ ((1, & 6), & 0) \\ ((17, & 19), & 1) \\ ((15, & 13), & 0) \\ ((19, & 9), & 0) \end{bmatrix}$$

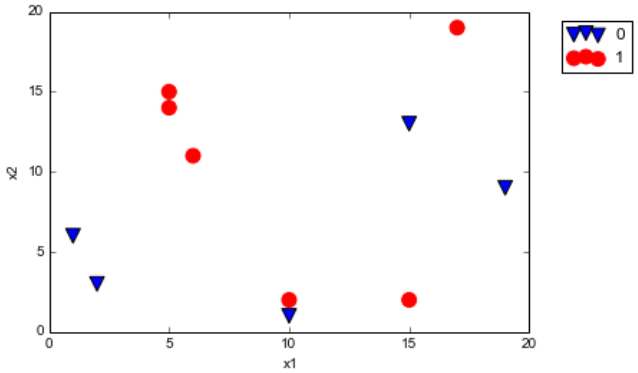


Figure 2: The data for Exercise 2.

## Exercise 3. $k$-Nearest Neighbors: Loss

(Based on slide 21)
Suppose a 7-nearest neighbors regression search returns $\{4, 2, 8, 4, 9, 11, 100\}$ as the 7-nearest $y$ values for a given $x$ value. Consider an hypothesis set made by two functions only: median and average. Let $\hat{y}$ be their prediction for $x$.

- If the evaluation is done by means of the *absolute value loss*,

  $$L_1 = |y - \hat{y}|$$

  which of the two functions minimizes the sum of absolute value loss function:

  $$\hat{L}_1 = \sum_{j=1}^{k} |y_j - \hat{y}|$$

  on the given data?

- If the evaluation is done by means of the *squared loss error*,

  $$L_2 = (y - \hat{y})^2$$

  which of the two functions minimizes the sum of squares loss function

  $$\hat{L}_2 = \sum_{j=1}^{k} (y_j - \hat{y})^2$$

  on the given data?

- [Advanced] Show that the average is the function that minimizes the loss function $L_2$ in a $k$-nearest neighbor method for regression on any data. [Hint: In $\hat{L}_2$, the only variable is $\hat{y}$ ($y_i$'s are given by the input). The value for $\hat{y}$ that minimizes $\hat{L}_2$ can be found by differentiation in $\hat{y}$.]

[Note: You can carry out the calculations by hand or you can use any program of your choice.]

## Exercise 4. Linear Regression: Prediction

(Based on slides 24-26)
As in Exercise 1. you are trying to predict a response $y$ to an input $x$ and you are given the same set of training data $D = [(x_1, y_1), \ldots, (x_{11}, y_{11})]$, also reported and plotted in Figure 3. However, now you want to use a linear regression model to make your prediction. After training, your model looks as follows:

$$g(x) = -0.37x + 11.22$$

The corresponding function is depicted in red in Figure 3. What is your prediction $\hat{y}$ for the new input $\hat{x} = 8$? What is the squared error loss if later you find out that the actual response for $\hat{x}$ is 9?

$$D = \begin{bmatrix} (8, & 8.31) \\ (14, & 5.56) \\ (0, & 12.1) \\ (6, & 7.94) \\ (3, & 10.09) \\ (2, & 9.89) \\ (4, & 9.52) \\ (7, & 7.77) \\ (8, & 7.51) \\ (11, & 8.0) \\ (8, & 10.59) \end{bmatrix}$$

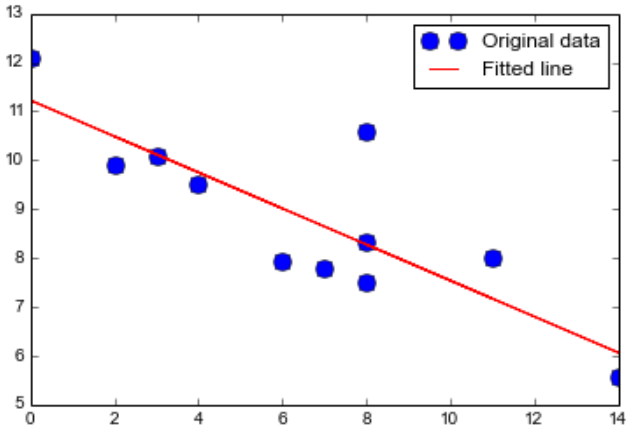


Figure 3: The data for Exercise 4.

## Exercise 5. Linear Regression: Training

(Based on Slides 24-26)
Calculate the linear regression line for the set of points:

$$D = \begin{bmatrix} (2, 2) \\ (3, 4) \\ (4, 5) \\ (5, 9) \end{bmatrix}$$

Calculate also the *training error* defined as the sum of squared errors on all data from $D$.
Plot the points and the regression line on the Cartesian coordinate system.
[You can carry out the calculations by hand or you can use any program of your choice. Similarly, you can draw the plot by hand or get aid from a computer program.]

## Exercise 6. Multilayer Perceptrons

(Based on Slides 66, 67.)
Determine the Boolean Function represented by the perceptron in Figure 4:

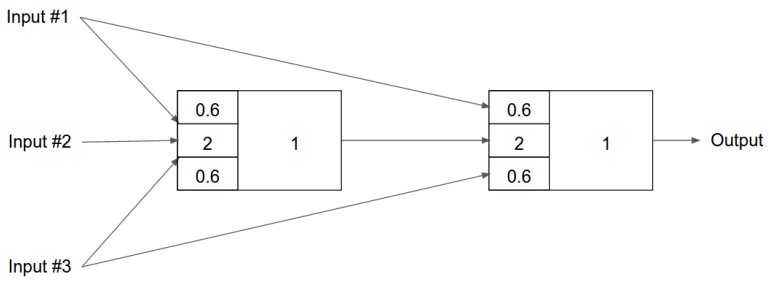


Figure 4: . The multilayer perceptron of Exercise 6.

## Exercise 7. Feed-Forward Neural Networks: Single Layer Perceptron

(Based on Slides 49-54)
Determine the parameters of a single perceptron (that is, a neuron with step function) that implements the majority function: for $n$ binary inputs the function outputs a 1 only if more than half of its inputs are 1.

## Exercise 8. Single Layer Neural Networks: Prediction

(Based on Slides 49-63.)
In Exercise 2. we predicted the class $y \in \{0, 1\}$ of an input $(x_1, x_2)$ with the 5-nearest neighbors method using the data from set $D$. We used those data to train a single layer neural network for the same task. The result is depicted in Figure 5.

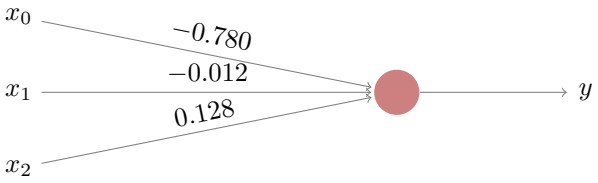


Figure 5: A single layer neural network for the task of Exercise 8.

- Calculate the prediction of the neural network for the new input $\hat{\vec{x}} = (5, 10)$. Assume a step function as activation function in the unit, which is therefore a perceptron.
- Calculate the prediction of the neural network for the new input $\hat{\vec{x}} = (5, 10)$. Assume a logistic function as activation function in the unit, which is therefore a sigmoid neuron.
- Compare the results at the previous two points against the result in Exercise 2. Are they all consistent? Which one is right?
- In binary classification, the training error can be defined as the number of mispredicted cases. Calculate the training error for the network under the two different activation functions. Which one performs better according to the training error?
- Derive and draw in the plot of Exercise 2. the decision boundaries between 0s and 1s that is implied by the perceptron and the sigmoid neuron. Are the points linearly separable? [See page 4 of Lecture Notes.]

## Exercise 9. Single Layer Perceptrons

(Based on Slides 49-67)
Can you represent the two layer perceptron of Figure 6 as a single perceptron that implements the same function? If yes, then draw the perceptron.

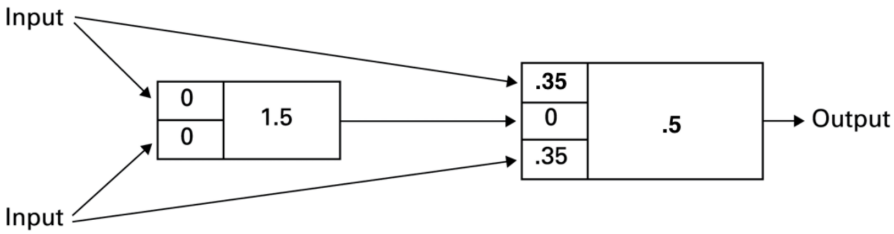


Figure 6: A two layer neural network

## Exercise 10. Expressivness of Single Layer Perceptrons

(Based on slide 63)
Is there a Boolean (logical) function in two inputs that cannot be implemented by a single perceptron? Does the answer change for a single sigmoid neuron?

## Exercise 11. Logical Functions and Neural Networks

(Based on Slide 49-63)
The NAND gate is universal for computation, that is, we can build any computation up out of NAND gates. We saw in the Study Group session that a single perceptron can model a NAND gate. From here, it follows that using networks of perceptrons we can compute any logical function.
For example, we can use NAND gates to build a circuit which adds two bits, $x_1$ and $x_2$. This requires computing the bitwise sum, $x_1$ XOR $x_2$, as well as a carry bit which is set to 1 when both $x_1$ and $x_2$ are 1, i.e., the carry bit is just the bitwise product $x_1 x_2$. The circuit is depicted in Figure 7.
Draw a neural network of NAND perceptrons that would simulate the adder circuit from the figure. [You do not need to decide the weights. You have already discovered which weights for a single perceptron would implement a NAND function in one of the exercises for the Study Group.]
What is the advantage of neural networks with respect to logical circuits?

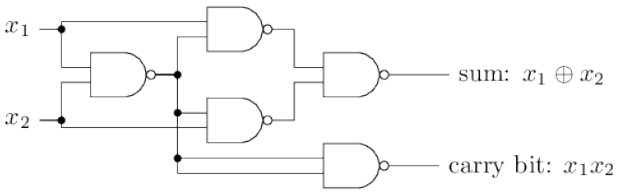


Figure 7: The adder circuit of Exercise 11.

## Exercise 12. Computer Performance Prediction

(Based on Slides 17-69)
You want to predict the running time of a computer program on any architecture. To achieve this task you collect the running time of the program on all machines you have access to. At the end you have a spreadsheet with the following columns of data:

(1) vendor name: 30 different brands

(2) MYCT: machine cycle time in nanoseconds (integer)

(3) MMIN: minimum main memory in kilobytes (integer)

(4) MMAX: maximum main memory in kilobytes (integer)

(5) CACH: cache memory in kilobytes (integer)

(6) CHMIN: minimum channels in units (integer)

(7) CHMAX: maximum channels in units (integer)

(8) Running time in seconds (integer)

Indicate which of the following is correct:

a. It is a supervised learning, regression task. Therefore, we can apply 5-nearest neighbors. For a new machine, the predicted running time in seconds is the average of the running time of the 5 closest machines. The distance between machines is calculated as the sum of the squares of the differences for each of the attributes from (1) to (7).

b. It is a supervised learning, regression task. Therefore, we can apply a linear model that takes attributes (1)-(7) as independent variables and attribute (8) as response variable.

c. It is a supervised learning, classification task. Therefore, we can train a multilayer neural network that has an input layer made by input nodes one for each of the attributes (1)-(7); an output layer made by one single sigmoid node that outputs the predicted running time in seconds; an hidden layer of say 10 nodes made by sigmoid nodes.

d. the same as in the previous point but where the output layer is now made by one single node whose activation function is a linear function. The task is then supervised learning, regression.

e. It is an unsupervised learning task.

f. It is a reinforcement learning task.

## Exercise 13. The Recursion Formula in Multilayer Neural Networks

(Based on Slide 66-67)
[Advanced] Suppose you have a single hidden layer neural network with *linear activation functions*. That is, for each unit the output is some constant $c$ times the weighted sum of the inputs plus a constant $d$.
For a given assignment of the weights $\vec{w}$, write down equations for the value of the units in the output layer as a function of $\vec{w}$ and the input layer $\vec{x}$, without any explicit mention of the output of the hidden layer. Show that there is a network with no hidden units that computes the same function.