Theoretical part of machine learning Ex1 Hodaya Koslowsky

1)

a) The instance domain X is all binary strings, where each bit represent “true” or “false”, and corresponds to a single x_i variable.

The label set Y is {0 or 1}, that represent for each example the boolean conjunction **result** of the |x| variables.

b) for d=2:

1. x_1 and x_2
2. not(x_1) and x_2
3. x_1 and not(x_2)
4. not(x_1) and not(x_2)
5. x_1
6. x_2
7. not(x_1)
8. not(x_2)
9. x_1 and x_2 and not(x_1) and not(x_2) (and other negative combinations)
10. 1

c) The formula is 3^d+1. For each variable x there are three options:

1. x appears.
2. not(x) appears
3. both x and not(x) do not appear.

Besides, there is one other option that is always negative – when for at least one variable both x and not(x) appear.

So we get three options to the power of d, plus the all negative option.

d) I. Yes it could. The fourth variable is not a part of the conjunction. And the example fits: not(1)*0*1 = 0

II. Yes, again, since x4 is not in the conjunction and can be assigned to either true or false without changing the result.

2)

a) Yes. First, lets define a loss function to be 1 if h(x)=0 and y=1, and 0 otherwise.

If the algorithm implements the ERM principle, it should find the h such that the average loss for all the examples is minimal.

We start with the all negative hypothesis, so $h^0(x) = 0$ for all x. The average loss function will be: (number of y=0)/(number of examples).

Then, for each example x, if h(x)=0 and y=1 we go through all the indexes of the example and delete the unfitting, making sure that for that example – the loss function will return 0.

If the dataset is consistent we should not get an example where h(x)=1 and y=0.

Therefore, we continue improving the hypothesis until the algorithm saw all the examples. By that time, it is fit to all of them and the average loss function will be 0.

b) We will prove that M(a) <= 2d:

Before we start h contains exactly 2d variables. (every variable and its negation).

If a mistake is found, meaning h(x)=0 and y=1, we remove at least one variable from h.

So after maximum 2d mistakes happen, h is empty and the algorithm must stop. Besides, in the first mistake, the algorithm deletes d variables: either xi or not(xi). So we have the first mistake, and then maximum d mistake, because at each mistake at least one variable is removed and there are only d variables left. So this gives us meximum d+1.

c) Each iteration takes O(d) time, because we go through all the indexes of the example.