Machine Learning- Exercise 1
Theoretical Part

Name: Tomer Gill (תומר גיל)
ID: 318459450
U2 Username: gilltom
Date: 19/03/2018

Part 1 - General definition of the problem

(a) $\mathcal{X} = \{(b_1, b_2, \dots, b_d) | b_i \in \{0,1\}, 1 \le i \le d\},\ \mathcal{Y} = \{0,1\}\ (for\ a\ given\ d)$

(b) $\mathcal{H} = \{x_1 \wedge x_2,\ \overline{x}_1 \wedge \overline{x}_2, x_1 \wedge \overline{x}_2, \overline{x}_1 \wedge x_2, x_1, x_2, \overline{x}_1, \overline{x}_2, \emptyset, x_1 \wedge x_2 \wedge \overline{x}_1 \wedge \overline{x}_2\}$

(c) $For\ a\ given\ size\ d{:}\ |\mathcal{H}| = 3^d + 1$

(d) For $\overline{x}_1 \wedge x_2 \wedge x_3$:

i. ((1,0,1,1),0) **CAN** exist in the training set because $\overline{1} \wedge 0 \wedge 1$ does equal 0. It means there are 4 literals (x_1, x_2, x_3, x_4) but the value of x_4 doesn't matter to the result (y).

ii. **Yes**, both ((0,1,1,0),1) and ((0,1,1,1),1) can exist in the training set, because x_4's value doesn't matter and they both fulfils the conjunction: $\overline{0} \wedge 1 \wedge 1 = 1$.

Part 2 - The Consistency Algorithm

(a) **Yes**, the algorithm implements the ERM principle, because it's "learning" is done using empirical data – it uses many examples' results to try to "learn" the distribution of the outputs.

(b) First, I'll prove by induction that $M(a) \le 2d$ and then I'll prove the bound is $d+1$:

a. **Base**: For $d = 0$, the algorithm makes $M(a) = 0 \le 2d = 2 * 0 = 0$ mistakes.
Step: Let's assume that for an arbitrary d, the algorithm makes $M(a) \le 2d$ mistakes, and we'll prove that for $d+1$, the algorithm makes $M(a) \le 2(d+1) = 2d+2$:
We have d+1 literals ($x_1, \dots, x_d, x_{d+1}$), so our initial hypothesis is the *all-negative hypothesis*: $\wedge_{i=1}^{d+1}(x_i \wedge \overline{x}_i) = \wedge_{i=1}^{d}(x_i \wedge \overline{x}_i) \wedge x_{d+1} \wedge \overline{x}_{d+1}$. The algorithm will go through our examples. The maximal number of mistakes that involve the literal x_{d+1} is 2: a mistake that will remove x_{d+1} and another mistake that will remove $\overline{x}_{d+1}$. In the "worst case", those will be removed from our hypothesis, and so we will be left with only d literals. From the induction's assumption we get that for d literals $M(a) \le 2d$.
So in total for d+1 literals, $M(a) \le 2d + 2 = 2(d+1)$.

b. If the real conjunction is the all negative hypothesis, then the algorithm will never make mistakes because it is right from the start. So, for all conjunctions that are not the all-negative, we have n literals or their negations – minimum 0, maximum of d ($0 \le n \le d$). The first time a mistake is made, then out of the $2d$ literals in the all-negative, at least d literals are

removed (for each literal there is a binary value in the example, and if a mistake is made then the appropriate literal/negation is removed) to a total of d [so far 1 mistake].
In the worst case, the real conjunction consists of only 1 literal and each mistake will only remove one literal from our hypothesis. So the maximal mistakes that can be made in the worst case is d mistakes.
For conclusion, we proved that maximal number of mistakes in the first mistake and then d mistakes, so $M(a) \leq 1 + d = d + 1$.

(c) In the worst case, an iteration is the first mistake – the algorithm goes over all d values of the example and update the hypothesis. Every other iteration goes over maximum of d literals.
So in total, an iteration's time is $O(d)$.