Prof. Friedrich Eisenbrand
Nicolai Hähnle

Discussions from: September 24, 2009

Mathematics of Machine Learning

Fall 2009

Assignment Sheet 1

Exercises marked with a $\star$ can be handed in for bonus points. Due date is October 8, 2009.

Exercise 1 ($\star$)
Generalize the algorithm for the rectangle learning game to prove that if $\mathscr{C}_n$ is the class of all axis-aligned hyperrectangles in n-dimensional Euclidean space $\mathbb{R}^n$, then $\mathscr{C}$ is efficiently PAC learnable.

Exercise 2
Consider the problem of learning boolean disjunctions. Here the instance space is $X_n = \{0,1\}^n$, where each $a \in X_n$ is interpreted as an assignment to the n boolean variables $x_1,\ldots,x_n$. The representation class $\mathscr{C}_n$ is the class of all disjunctions of literals over $x_1,\ldots,x_n$. For example, the disjunction $x_1 \vee \bar{x}_2 \vee x_4$ represents the set $\{a \in \{0,1\}^n \mid a_1 = 1 \text{ or } a_2 = 0 \text{ or } a_4 = 1\}$.

Show that the representation class of disjunctions of boolean literals is efficiently PAC learnable.

Exercise 3
We modified the PAC model to allow the learning algorithm time polynomial in n (the "size" of the instances) and $size(c)$ (the "size" of the target concept) and we also provided the value $size(c)$ as input. Prove that this input is actually unneccessary: if there is an efficient PAC learning algorithm for $\mathscr{C}$ that is given $size(c)$ as input, then there is an efficient PAC learning algorithm for $\mathscr{C}$ that is not given this input.

Addendum: You should run into difficulties when trying to prove that your algorithm always runs in polynomial time. When you do so, try to find an an algorithm with a weaker condition on the running time: Can you find an algorithm that, with probability at least $1-\delta$ runs in polynomial time, but might not terminate otherwise? Can you find an algorithm that has expected polynomial running time?

Exercise 4
Recall the Independent Set (or Stable Set) problem:

> Given an undirected graph $G = (V,E)$ and an integer k, determine if there exists a set $S \subseteq V$ of at least k vertices that are not adjacent in G (such a set S is called *independent* or *stable*).

and the Vertex Cover problem:

> Given an undirected graph $G = (V, E)$ and an integer k, determine if there exists a set $C \subseteq V$ of at most k vertices such that every edge in E has at least one endpoint in C.

Show that these problems can be reduced to each other, that is: given an instance I of the Independent Set problem, you can construct an instance I' of the Vertex Cover problem such that the answer to I is YES if and only if the answer to I' is YES, and vice versa.

Exercise 5
Recall the Graph 3-Coloring problem:

> Given an undirected graph $G = (V, E)$, determine if there is an assignment $c : V \to \{1, 2, 3\}$ of at most three different colors to the vertices of G such that for every edge $\{i, j\} \in E$, vertex i and j are assigned different colors.

Show that the Graph 3-Coloring problem is NP-complete.

Hint: You may use the fact that 3-SAT is NP-complete. Recall that 3-SAT is the following problem:

> Given a 3-CNF formula, that is, a formula of the form $\bigwedge_{i \in I}(u_i \vee v_i \vee w_i)$ where u_i, v_i, w_i are literals, decide whether there exists an assignment to the boolean variables such that the formula is satisfied.

Hint #2 (given in the exercise session): When constructing the graph from the 3-CNF formula, design it such that the three colors in a 3-coloring correspond to True, False, and Helper. Introduce vertices for literals x_i and $\bar{x}_i$ and make sure they are colored correctly. Finally, you will need to come up with some addition structures to represent the clauses.

Exercise 6
Recall that the representation class of 3-CNF formulae is efficiently PAC learnable. Is the representation class of CNF formulae – that is, formulae where each clause can contain arbitrarily many literals – also efficiently PAC learnable?

Exercise 7 ($\star$)
Consider the following *two-oracle* variant of the PAC model: when $c \in \mathscr{C}$ is the target concept, there are separate and arbitrary distributions $\mathscr{D}_c^+$ over only the positive examples of c and $\mathscr{D}_c^-$ over only the negative examples of c. The learning algorithm must find a hypothesis h satisfying $\Pr_{x \in \mathscr{D}_c^+}[h(x) = 0] \leq \varepsilon$ and $\Pr_{x \in \mathscr{D}_c^-}[h(x) = 1] \leq \varepsilon$. In other words, the learning algorithm may now explicitly request either a positive or a negative example, but must find a hypothesis with small error on both distributions.

Let $\mathscr{C}$ be any concept class and $\mathscr{H}$ be any hypothesis class. Let h_0 and h_1 be representations of the identically 0 and identically 1 functions, respectively. Prove that $\mathscr{C}$ is efficiently PAC learnable using $\mathscr{H} \cup \{h_0, h_1\}$ in the original one-oracle model if and only if $\mathscr{C}$ is efficiently PAC learnable using $\mathscr{H}$ in the two-oracle model.