INF367 - Selected Topics in AI

Learning Theory and Neuro-symbolic AI

Assignment 1

August 23, 2020

This assignment contributes to 6% of the final grade. Your grade will be based on the clarity and correctness of the results. You may talk with your colleagues but you should understand yourself all the steps of your answers/code and write with your own words. Copying is not allowed.

Deliverables:

- A jupyter notebook containing all the code to reproduce your work and
- a report explaining your answers/code.

1 PAC Learning (Points 1)

1. Let $F = (E, H, m)$ be the learning framework where
   - H is the set of all conjunctions of literals that can be formulated in $V = v_1, \ldots, v_n$;
   - E is the set of all valuations in V;
   - m is a function that maps each conjunction of literals in H to the valuations that satisfy it.

   Write *with your own words* all the steps for showing that F is PAC learnable in polynomial time.

2. Write *with your own words* all the steps for showing that, for all learning frameworks $F = (E, H, m)$, if H is finite then F is PAC learnable.

2 Hands on Code (Points 5)

Implement an algorithm for PAC learning the learning framework presented in Item 1 of Section 1 (conjunctions of literals) and an "oracle" that returns classified examples according to a probability distribution. Your program should also be able to estimate the error of the hypothesis.

Input: The parameters ϵ (upper bound for the error of the hypothesis) and δ (where $1 - \delta$ is the confidence of the hypothesis). The parameters to define the probability distribution (for a normal distribution, these are mean and covariance).

Guidelines: These are guidelines for the implementation.

- How to generate the examples randomly:

  You can use a normal distribution to generate your sample.

- How to select the target:

  You can generate the target randomly or fix one in your code (suggestion: use $n = 10$ variables).

- How to select the hypothesis:

  Generate a set of examples of size greater or equal to

  $$(2n)/\epsilon(ln(2n) + ln(1/\delta))$$

  where n is the number of variables occurring in your target. Use the target to label the examples and create a hypothesis consistent with the examples by applying the strategy in the slides of Lec 2 (also in Chapter 1 of Introduction to Computational Learning Theory).

- How to estimate the error of the hypothesis:

  The error of a hypothesis h w.r.t. a target t and a probability distribution D—in symbols, $D(m(t) \oplus m(t))$—is difficult to compute. You can estimate it as follows. Pick a set S of examples of size N generated according to D. Calculate $\hat{t} = \frac{|S \cap (m(t) \oplus m(h))|}{N}$. It can be shown that $\hat{t}$ is an unbiased estimator of $D(m(t) \oplus m(h))$. Choose $N = (20/\epsilon)^2$.

Output: The hypothesis and an estimate of the error of the hypothesis w.r.t. the target and the probability distribution used for generating the training set.

Generalization guarantee: Given a training set of size

$$(2n)/\epsilon(ln(2n) + ln(1/\delta))$$

with probability at least $1 - \delta$ the error of the hypothesis should be bounded by ϵ. You program should be able to verify this guarantee by running the algorithm $10/\delta$ times and showing that in at most 10 out of $10/\delta$ times the error of the hypothesis is greater than ϵ.