Aalto University
School of Science

CS-E4070 — Computational learning theory

Slide set 11 : online learning

Cigdem Aslay and Aris Gionis

Aalto University

spring 2019

reading material

- Nick Littlestone, “Learning Quickly When Irrelevant Attributes Abound – A New Linear-threshold Algorithm.” Machine Learning, 1987

overview

- mistake-bound model
  - basic results, the HALVING algorithm
  - connections to information theory
  - the WINNOW algorithm

recap — PAC learning

- $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ where $\mathbf{x}$ is sampled from $\mathcal{D}$, and $y = c(\mathbf{x})$ labeled by the target concept $c : X \to Y$ that we want to learn
- the learner observes sample set S and outputs hypothesis $h : X \to Y$ for predicting the label of unseen data points drawn from $\mathcal{D}$.
- the error of the learner is defined as the probability that the learner does not predict the correct label on a random data point sampled from $\mathcal{D}$

$$error_{\mathcal{D}}(h) = \mathbf{Pr}_{\mathbf{x} \sim \mathcal{D}}[h(\mathbf{x}) \neq c(\mathbf{x})]$$

online learning

- assumption in PAC learning:
  - error is measured on a fixed distribution
  - same distribution used to learn the hypothesis
- what if we do not want to make this assumption ?
  - cannot make claims about predicting future results
- can we say anything interesting ?

  mistake bounds and regret bounds

mistake-bound model

- view learning as an iterative process
- in each iteration
  - algorithm is given $\mathbf{x}$
  - predicts $h(\mathbf{x})$
  - told the true label $c(\mathbf{x})$, and if made a mistake
- no assumptions about order of examples or distribution
- objective: bound the total number of mistakes

mistake-bound model

- definition: algorithm A learns concept class $\mathcal{C}$ with mistake bound M if A makes at most M mistakes on any sequence of examples consistent with some $c \in \mathcal{C}$
- note: we can no longer talk about total number of examples required to learn a hypothesis
  - maybe we see the same examples over again and learn nothing new
  - but this is OK if do not make mistakes
- want mistake bound $poly(n, s)$, where n is size of example and s is size of smallest consistent $c \in \mathcal{C}$

mistake-bound model

- definition: a concept class $\mathcal{C}$ is learnable in the MB model if there exists an algorithm A whose mistake bound and running time per iteration is $poly(n, s)$

example : boolean disjunctions

- consider n boolean variables $x_1, \ldots, x_n$
- concept class: boolean monotone disjunctions
  - e.g., $c(\mathbf{x}) = x_1 \vee x_3 \vee x_4 \vee x_9$
  - no negations
- can we learn target concept with at most n mistakes ?
- online learning algorithm:
  - start with $h(\mathbf{x}) = x_1 \vee x_2 \vee \ldots \vee x_n$
  - invariant: $\{\text{variables in } c\} \subseteq \{\text{variables in } h\}$
  - mistake on positive example: do nothing
  - mistake on negative example: remove x_i's set to 1
- analysis: invariant is maintained
- for each mistake we remove at least one variable:
  - we cannot remove more than n variables

example : boolean disjunctions

- the online learning algorithm makes at most n mistakes
- any algorithm can be forced to make at least n mistakes

$$\begin{array}{cccccc} 1 & 0 & \ldots & 0 & & + \text{ or } - \\ 0 & 1 & \ldots & 0 & & + \text{ or } - \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \ldots & 1 & & + \text{ or } - \end{array}$$

MB model properties

- an algorithm A is conservative if it only changes its state when it makes a mistake
- claim: if $\mathcal{C}$ is learnable by a deterministic algorithm with mistake bound M, then it is learnable by a conservative algorithm with mistake bound M
- why ?

MB learnability implies PAC learnability

- consider online learning algorithm A with mistake bound M
- transformation:
  - run (conservative) A until it produces a hypothesis h that survives at least $(1/\epsilon)\ln(M/\delta)$ examples
- $\Pr[\text{fooled by a given "bad" hypothesis}] \leq \delta/M$
- $\Pr[\text{fooled by any "bad" hypothesis}] \leq \delta$
- total number of examples seen is at most $(M/\epsilon)\ln(M/\delta)$

for details see [Kearns et al., 1987]

see also homework question

what if we had unbounded computational power ?

- consider the HALVING algorithm
  - an analogue of binary search
- maintain the version space: the set of all concepts that are consistent with all examples seen so far
- more formally
  - CONSISTENT $= \{c \in \mathcal{C}$ s.t. c consistent with previous examples $\}$
  - for instance $\mathbf{x}$ and concept class $\mathcal{C}$:

$$\xi_0(\mathcal{C}, \mathbf{x}) = \{c \in \mathcal{C} \mid c(\mathbf{x}) = 0\}$$

$$\xi_1(\mathcal{C}, \mathbf{x}) = \{c \in \mathcal{C} \mid c(\mathbf{x}) = 1\}$$

HALVING algorithm

- $\text{CONSISTENT} = \mathcal{C}$
- upon seen instance $\mathbf{x}$
  - if $|\xi_1(\text{CONSISTENT}, \mathbf{x})| > |\xi_0(\text{CONSISTENT}, \mathbf{x})|$

    predict 1
  - if $|\xi_1(\text{CONSISTENT}, \mathbf{x})| \leq |\xi_0(\text{CONSISTENT}, \mathbf{x})|$

    predict 0
  - if correct label is 1

    $\text{CONSISTENT} = \xi_1(\text{CONSISTENT}, \mathbf{x})$
  - if correct label is 0

    $\text{CONSISTENT} = \xi_0(\text{CONSISTENT}, \mathbf{x})$

HALVING algorithm

- theorem: the number of mistakes of the HALVING algorithm is bounded by $\log |\mathcal{C}|$

what if we had unbounded computational power ?

- what if we had a prior p over concepts of $\mathcal{C}$?
  - weight the vote according to p
  - make at most $\log(1/p_c)$ mistakes,

    where c is the target concept
- what if c was really chosen according to p ?
  - expected number of mistakes $\leq \sum_c p_c \log(1/p_c)$

    the entropy of the distribution p

the WINNOW algorithm

- online learning of monotone boolean disjunctions
  - mistake bound: n
- can we do better ?
- assume that disjunction contains at most k literals
  - e.g., $c(\mathbf{x}) = x_{i_1} \vee \ldots \vee x_{i_k}$, for $k << n$
- well-motivated assumption: in many applications only a small number of variables is relevant

winnow [win-oh] SHOW IPA

EXAMPLES | WORD ORIGIN

SEE MORE SYNONYMS FOR *winnow* ON THESAURUS.COM

verb (used with object)

1 to free (grain) from the lighter particles of chaff, dirt, etc., especially by throwing it into the air and allowing the wind or a forced current of air to blow away impurities.

2 to drive or blow (chaff, dirt, etc.) away by fanning.

the WINNOW algorithm

- the algorithm is applicable to learning binary functions $c : \{0, 1\}^n \to \{0, 1\}$ that are linearly separable
  - i.e., there is a hyperplane that separates positive from negative instances
- e.g., monotone disjunction $c(\mathbf{x}) = x_1 \vee x_3 \vee x_4 \vee x_9$ is linearly separable
  - why ? consider hyperplane

$$x_1 + x_3 + x_4 + x_9 = 1/2$$

the WINNOW algorithm

- maintain weights $w_1, \ldots, w_n$ associated with variables $x_1, \ldots, x_n$
- initially $w_1 = \ldots = w_n = 1$
- use parameters θ and α
- to predict label of instance $(x_1, \ldots, x_n)$ use the rule:
  - if $\sum_i w_i x_i > \theta$ predict 1
  - if $\sum_i w_i x_i \leq \theta$ predict 0
- weights $w_1, \ldots, w_n$ are updated when algorithm makes a mistake
  - weights update is controlled by parameter α

WINNOW's response to mistakes

learner's prediction	correct response	update action	response name
1	0	$w_i = 0$ if $x_i = 1$ w_i unchanged if $x_i = 0$	elimination step
0	1	$w_i = \alpha w_i$ if $x_i = 1$ w_i unchanged if $x_i = 0$	promotion step

WINNOW's performance

- theorem: assume that the target concept is a k-literal monotone disjunction $c(x_1, \ldots, x_n) = x_{i_1} \vee \ldots \vee x_{i_k}$
  If WINNOW is run with $\alpha > 1$ and $\theta > 1/\alpha$, then for any sequence of instances the total number of mistakes will be bounded by
  $$\alpha k(\log_\alpha \theta + 1) + \frac{n}{\theta}$$

WINNOW's performance

- mistake bound:

$$\alpha k(\log_\alpha \theta + 1) + \frac{n}{\theta}$$

- if $\theta = n$ and $\alpha = 2$, bound is $2k(\log_2 n + 1) + 1$
- if $\theta = n/\alpha$, bound is $\alpha k \log_\alpha n + \alpha$
- if $\theta = n/2$ and $\alpha = 2$, bound is $2k \log_2 n + 2$

analysis of the WINNOW algorithm

- theorem: assume that the target concept is a k-literal monotone disjunction $c(x_1, \ldots, x_n) = x_{i_1} \vee \ldots \vee x_{i_k}$
  If WINNOW is run with $\alpha > 1$ and $\theta > 1/\alpha$, then for any sequence of instances the total number of mistakes will be bounded by
  $$\alpha k(\log_\alpha \theta + 1) + \frac{n}{\theta}$$

- **proof**

analysis of the WINNOW algorithm

- lemma 1: let p be the number of promotion steps;
  let e be the number of elimination steps; then:

$$e \leq \frac{n}{\theta} + (\alpha - 1)p$$

proof

- initially $\sum_i w_i = n$
- each promotion increases the sum by at most $(\alpha - 1)\theta$
  - because promotion happens when $\sum_i w_i x_i \leq \theta$
- each elimination decreases the sum by at least θ
- since the sum is never negative we have

$$0 \leq \sum_i w_i \leq n + \theta(\alpha - 1)p - \theta e$$

analysis of the WINNOW algorithm

- lemma 2: $w_i \leq \alpha\theta$, for all i

  proof

- since $\theta > 1/\alpha$ the condition initially holds
- weight w_j is increased only if $\sum_i w_i x_i \leq \theta$ and $x_j = 1$
  - thus, before promotion $w_j \leq \theta$
  - thus, after promotion $w_j \leq \alpha\theta$

analysis of the WINNOW algorithm

- lemma 3: after p promotion steps and an arbitrary number of elimination steps there exists some i s.t., $\log_\alpha w_i \geq p/k$

proof

- let $R = \{x_{i_1}, \ldots, x_{i_k}\}$ and consider $\prod_{i \in R} w_i$
- $c(x_1, \ldots, x_n) = 0$ if and only if $x_i = 0$ for all $x_i \in R$
- elimination occurs when $c(x_1, \ldots, x_n) = 0$
  - elimination lefts $\prod_{i \in R} w_i$ unchanged
- promotion occurs when $c(x_1, \ldots, x_n) = 1$
  - promotion increases $\prod_{i \in R} w_i$ by at least α
- after p promotion steps $\prod_{i \in R} w_i \geq \alpha^p$
- by PHP, there exists some i s.t., $\log_\alpha w_i \geq p/k$

analysis of the WINNOW algorithm

proof of theorem

- number of mistakes is equal to $p + e$
- by lemmas 3 and 2, there exists some i s.t.,

$$p/k \leq \log_\alpha w_i \leq \log_\alpha \theta + 1$$

or

$$p \leq k(\log_\alpha \theta + 1) \tag{1}$$

- by lemma 1

$$e \leq \frac{n}{\theta} + (\alpha - 1)p \leq \frac{n}{\theta} + (\alpha - 1)k(\log_\alpha \theta + 1) \tag{2}$$

- (1)+(2) gives the result

analysis of the WINNOW algorithm

- **lower bound**: the number of mistakes required to learn a k-literal monotone disjunction is at least $\frac{k}{8}(1 + \log_2 \frac{n}{k})$

summary of the course

- introduction to PAC learning model
- Occam's razor
- agnostic learning
- VC dimension
- weak and strong learning, and boosting
- learning in the presence of noise: statistical query learning
- submodular optimization and applications
- online learning: mistake-bound models

some topics we did not manage to cover

- Rademacher complexity and covering numbers
- online learning: regret bounds
- randomized weighted majority algorithm

references

Kearns, M., Li, M., Pitt, L., and Valiant, L. G. (1987).
Recent results on boolean concept learning.
In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 337–352. Elsevier.