Lesson 7: Computational Learning Theory

Outline

- Defining learning problems.
- Showing specific algorithms work.
- Showing these problems are fundamentally hard.

Resources in machine learning

- Time
- Space
- Training samples

Defining inductive learning

1. Probability of successful training: $1 - \delta$
2. Number of training examples: m
3. Complexity of hypothesis class H
4. Accuracy to which target concept is approximated: ϵ
5. Manner in which training examples presented. (especially matters in *online training* as opposed to *batch*)
6. Manner in which training examples selected.

Selecting training examples

How learner/teacher interact.

1. Learner asks questions of teacher: $x \rightarrow c(x)$
2. Teacher gives examples to help learner: x and $c(x)$
3. Some natural fixed distribution: x chosen from D by nature.
4. Evil distribution: ask questions in an evil way.

Teaching via 20 questions

H – set of possible people (ask the correct question)

X – set of questions

- Teacher chooses only one question (he knows the right answer).
- Learner chooses $log|H|$ questions.

Teacher with constrained queries

H: Conjunction of literals or their negations

X: $x_1x_2x_3...x_k$ k-bit input

To solve: * Show what is irrelevant… two positive examples which are unperturbed by a changing feature. * Show what is relevant… k negative examples which validate that perturbing a relevant feature matters.

Even though there are 3^k (positive, absent, negated for each bit) hypotheses, the smart teacher can ask $k+2$ questions.

Learner with constrained queries

H: Conjunction of literals or negation

X: $x_1x_2x_3...x_k$ k-bit input

Learner does not know the actual answer like the teacher does so does not know what the right training examples are.

The ideal case is to have $log_2 3^k = klog_2 3$ (linear) time to get the concept, But due to the learner don't know the answer and the questions to be asked are **constrained** (cannot ask like "should x_1 be positive or negative"). So it can only start **enumerating** every data point from 0,…,0 to 1,…,1 which is 2^k possibilities.

The first positive result helps us significantly. But it actually takes $O(2^k)$ time to get there. Before that, most queries cannot give any information.

Learner with mistake bounds

Total number of mistakes will not be larger than a certain amount.

H: Conjunction of literals or negation – but not negation of conjunction? (find the right conjunction)

X: $x_1x_2x_3...x_k$ k-bit input

- Input arrives
- Learner guesses answer
- Wrong answer **charged**
- Go to 1.

Algorithm (We have a hypothesis, in such case, which is a table indicates the role of each bit. We can compute the output from the input based on the hypothesis):

1. Assume each feature both positive and negated in the table.
2. Given input, compute output according to our table.

3. If the computation from table is wrong, set all positive features that were 0 to absent, negative features that were 1 to sbsent. Go to (2).

Never make more than $k + 1$ mistakes from the hypothesis.

Definitions

- Learner chooses examples
- Teacher chooses examples
- **Nature chooses examples**
- (Mean teacher chooses examples)

Computational complexity – how much computational effort is needed for learner to "converge"

Sample complexity – batch; how many training examples are needed for a learner to create a successful hypothesis

Mistake bounds – online; how many misclassifications can a learner make over an infinite run?

Version spaces

- True/target hypothesis (concept): $c \in H$
- Candidate hypothesis: $h \in H$
- Training set: $S \subseteq X$
- **Consistent learner** : produces h such that $c(x) = h(x)$ for $\forall x$ in S (always produces a hypothesis that is consistent with data)
- **Version space**: $VS(S) = \{h \in H \ s.t. \ h \ is \ consistent \ with \ S\}$ – hypotheses consistent with examples.

PAC (Probably Approximately Correct) learning

Error of hypothesis h

Training error – fraction of traning examples misclassified by h.

True error – fraction of examples that would be misclassified on sample drawn from distribution D.

$error_D(h) = Pr_{x \sim D}[c(x) \neq h(x)]$

Concepts

- c – concept class

- L – learner
- H – hypothesis space
- n – $|H|$, size of hypothesis space
- D – distribution over inputs
- $0 <= \epsilon <= 1/2$ (our error goal: true error produced by the hypothesis is no bigger than ϵ)
- $0 <= \delta <= 1/2$ (failure-probability – certainty goal – with probability $1 - \delta$, must produce true error less than ϵ)

PAC – probably $(1 - \delta)$ approximately (ϵ) correct $(error_D(h) = 0)$!

Definitions

C is PAC-learnable by L using H iff learner L will with probability $1 - \delta$, output a hypothesis h in H such that $error_D(h) <= \epsilon$ in time and samples polynomial in $1/\epsilon$, $1/\delta$, and n.

Quiz: PAC-learnable

$C = H = h_i(x) = x_i$ (k-bit inputs)

There are k hypotheses. So $n = |H| = k$.

Pick a hypothesis uniformly from $VS(S, H)$.

ϵ-exhausted version space

$VS(S)$ is ϵ-exhausted iff for all h in $VS(S)$, $error_D(h) <= \epsilon$

Every element in the version space has error less than epsilon.

Haussler Theorem – bound true error

Let $error_D(h_1, ..., h_k \in H) > \epsilon$ – This indicates these k hypotheses have high true error.

How much data do we need to knock out these hypotheses?

$$Pr_{x \sim D}(h_i(x) = c(x)) \leq 1 - \epsilon$$

$$Pr(h_i \text{ consistent with } c \text{ on } m \text{ examples }) \leq (1 - \epsilon)^m$$

$Pr(\text{at least one of } h_1, ..., h_k \text{ consistent with } c \text{ on } m \text{ examples}) \leq k(1-\epsilon)^m \leq |H|(1-\epsilon)^m$

Note that $(1-\epsilon)^m \leq exp(\epsilon * m)$ (this comes from $-\epsilon \geq ln(1-\epsilon)$)

So we have:

$Pr(\text{at least one of } h_1, ..., h_k \text{ consistent with } c \text{ on } m \text{ examples}) \leq |H|(1-\epsilon)^m \leq |H|e^{-\epsilon m}$

This is the upperbound that version space is **not** ϵ-exhausted after m samples.
We want δ to be a bound on this. i.e.

$$(ln|H|)(-\epsilon * m) <= ln\delta$$

$$\Rightarrow m \geq \frac{1}{\epsilon}(ln|H| + ln(\frac{1}{\delta}))\ (i.e. polynomial)$$

Satisfying this will satisfy **PAC-learnability**.

Quiz: PAC-learnable example

$H = h_i(x) = x_i$ (where x is 10-bits)

$\epsilon = 0.1$ $\delta = 0.2$ D: uniform

$1/0.1 * (ln10 + ln(1/0.2)) = 10 * (ln10 + ln5) = 10 * ln50 = 39$

This bound is agnostic to distribution of nature's data.

Summary

- Teachers versus learners and interaction
  - What is learnable? Like complexity theory for ML.
- Sample complexity - data
- types of interactions:
  - learner picks questions
  - teacher picks questions
  - nature picks questions
  - evil teacher picks questions
- mistake bounds (as opposed to how many samples you need)
- PAC learning: version spaces, training/test/true error, distribution.
- $m \geq \frac{1}{\epsilon}(ln|H| + ln(\frac{1}{\delta}))$

- this assumed target in hypothesis
- otherwise **agnostic** and the bound is slightly different, but still polynomial.
- infinite hypothesis space can be a problem