Today’s Topic

Review: Occam’s Razor

There is a constant $b > 0$, such that for any concept class $C_n$ over $n$ attributes and any distribution $D$ on $n$ attribute examples, if the number of examples $m$ is such that $B(n, m) \leq b m - \left(\log_2 \frac{1}{\delta} - 1\right)$ and $m \geq \frac{b}{b^2} B(n, m) + \log_2 \frac{1}{\delta} + 1$, then with probability $1 - \delta$ any solution $h$ to $cons_{B}^{H}$ on the data set satisfies $Pr_{x \in D}[c(x) = h(x)] \geq 1 - \epsilon$.

Topics

1. Learning Decision Lists
2. Agnostic Learning

1 Learning Decision Lists

Definition 1 A decision list is a decision tree in which every internal node has at least one leaf as a child.

\[
\begin{array}{cccccccc}
  & x_2 & 0 & \neg x_1 & 0 & \neg x_3 & 0 & \neg x_4 & 0 & 1 \\
  \downarrow & 1 & 1 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow \\
  & 1 & 1 & 0 & 0 & \\
\end{array}
\]

In the above decision list, 1000 evaluates to 0; 1010 also evaluates to 0.

Equivalently, a decision list is of the form:

1: if $l_1$ then
2: return $b_1$
3: else if $l_2$ then
4: return $b_2$
5: ... 
6: else if $l_n$ then
7: return $b_n$
8: end if

We only need to find a consistent decision list, and it only need ever contain at most $n$ nodes. Since we can write each node using $O(\log n)$ bits, $B(n, m) \leq O(n \log n)$, using Occam’s razor.

If $m$ is the number of examples, then Occam’s razor ensures that if $m \geq \Omega(\frac{1}{\epsilon}(n \log n + \log \frac{1}{\delta}))$ we will find a list with an error of at most $\epsilon$ with probability $1 - \delta$.

The list only needs at most $n$ nodes because there’s no need to read a literal more than once.

Definition 2 We first fix some notation for a literal $l$ and set of labeled examples $S$. Let $S_l = \{(x, b) \in S : l(x) = 1\}$. We then say that $l$ is useful for $S$ if:
1. $S_l$ is nonempty

2. Every $(x, b) \in S_l$ has the same label (where $b$ equals either 0 or 1)

1.1 Algorithm

1: function FINDLIST($S$)
2: Let $h$ be the empty list.
3: repeat
4: Iterate over the literals until a useful literal is found.
5: For the label $b$ of those examples in $S_l$:
6: add $l$ to $h$.
7: Set $S \leftarrow S - S_l$.
8: until every $(x, b) \in S_l$ has the same label
9: Return $h$
10: end function

1.2 Correctness

It suffices to show that the algorithm always finds a useful literal – indeed, then every example in $S$ is initially removed at the stage where a leaf is added that labels it correctly.

If a useful literal always exists, then repeated application of the algorithm will always consume the entire set of literals.

Proof that a useful literal always exists:

Proof Consider the first literal $l$ of the target $c$ that is not tested by $h$, and consider the set of examples that are labeled by this literal $l$ in $c$ – let this set be $S'_l$. Since all previous literals are tested by $h$, at this stage $S_l \subseteq S'_l$. Thus, every example in $S_l$ is labeled by the same branch of $c$, and $l$ is useful at this point.

Once all of the literals of $c$ are added to the list, the rest of the examples are labeled by the terminal leaf of $c$, and hence receive the same label.

We recall that decision lists are the special cases of decision trees, which captured conjunctions and disjunctions. Much as how we extended the Elimination Algorithm to learn k-DNFs, we can also extend the decision list algorithm to learn “k-decision lists”, where instead of a single literal, each internal node tests a conjunction of size $\leq k$ literals by searching for “useful” conjunctions of size $\leq k$ instead of simply literals.

The overall running time of the algorithm is then $O(mn^{2k})$, which is still polynomial in the relevant parameters.

2 Agnostic Learning

Last time, we proved a formalization of “Occam’s Razor” - any algorithm solving $\text{consis}_B^H$ for data labeled by $c \in C$ returns an $h \in H$ such that $Pr_{x \in D}[c(x) = h(x)] \geq 1 - \epsilon$ with probability $1 - \delta$ as long
as it was given $m \geq \frac{\ln 2}{\epsilon}(B(n, m) + \log_2 \frac{1}{\delta} + 1)$ examples as input.

Suppose that computation time is not an issue, and $B(n, m)$ grows sublinearly in $m$ for fixed $n$. Describe an algorithm that finds an $h \in H$ such that, with probability $1 - \delta$, $Pr_{x \in D}[c(x) = h(x)] \geq 1 - \epsilon$ whenever the data is labeled by some $h \in H$.

Once $n$, $\epsilon$, and $\delta$ are fixed, if $B(n, m) = o(m)$ then, for sufficiently large $m$, $B(n, m) \leq \frac{1}{\ln^2} \epsilon m - \log_2 \frac{1}{\delta} - 1$. We compute such an $m$ and, using $m$ examples, enumerate $h \in H$ of length $\leq B(n, m)$ until we find one that correctly labels the data. Occam’s Razor then guarantees that this $h$ generalizes.