VC dimension and PAC Learning

1 Review

Denote the concept class as $C$ and a specific concept as $c \in C$. Let $X$ be the sample space and $S$ be a sample from $X$. We define a set of all possible behaviors on $S$ w.r.t. $C$, $\Pi_C(S)$ as $\{c \cap S | c \in C\}$, and define $\Pi_C(m) = \max_{|S|=m} |\Pi_C(S)|$. If $|\Pi_C(S)| = 2^{|S|}$, then we say that $C$ shatters $S$. Thus, the Vapnik-Chervonenkis dimension of $C$, denoted as $VCD(C)$ is the size of the largest shattered set; $\infty$, if $C$ can shatter sets of any size.

For example, no interval can realize the following labeling for the following sample:

Thus $|\Pi_C(S)| < 2^{|S|}$ and the interval concept class doesn’t shatter this sample.

In order to prove $VCD(C) = k$ for some concept class $C$, we can show $VCD(C) \geq k$ and $VCD(C) < k + 1$ as follows:

- To show $VCD(C) \geq k$: prove $\exists S$ of size $k$, $C$ shatters $S$.
- To show $VCD(C) < k + 1$: prove $\forall S$ of size $k + 1$, $C$ doesn’t shatter $S$.

For example, in order to prove the $VCD$ of axis-aligned rectangles in the plane is 4 we can argue as follows:

1. To show $VCD(C) \geq 4$: find one set of 4 points and then prove for each of the $2^4 = 16$ different labelings, we can find one rectangle that realizes the respective labeling. This can be done by firstly choosing one particular set of 4 points and then realizing all 16 labelings via rectangles.

2. To show $VCD(C) < 5$: pick any set of size 5, calculate $X_{\text{min}}$, $X_{\text{max}}$, $Y_{\text{min}}$ and $Y_{\text{max}}$, and then for each of the four boundaries pick any point that is at the boundary. Thus we select at most 4 points, which we denote as $A \subset S$. We consider the labeling where all points in $A$ are positive and points in $S \setminus A$ are negative. Any rectangle that includes $A$ as positive examples must also include $S \setminus A$ as positive and therefore this labeling cannot be realized. By showing that for any choice of points no rectangle can implement this mapping, we proved that the concept class can’t shatter any set of 5 points.

2 Motivation

The Occam Theorem provides an upper bound on the sample complexity of PAC learning $O\left(\frac{1}{\epsilon} \ln \frac{|H|}{\delta}\right)$. This is obviously not useful for infinite classes. In this and the next lecture we will show that similar
results can be stated for such classes by using the notion of VC dimension. Roughly speaking, we replace the “complexity of concept class” \(\ln |H|\) with the VC dimension. In fact, denoting \(d = VCD(C)\), we will prove both an upper bound \(O(\frac{d}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\delta})\) and a reasonably tight lower bound \(\Omega(\frac{d}{\epsilon})\) on the number of samples required for PAC learning.

3 Upper Bound on Sample Size

The upper bound is discussed in the next lecture. Here we prepare the ground for this by developing some combinatorial properties related to the VCD.

**Definition 1** For any natural numbers \(m\) and \(d\), the function \(\Phi_d(m)\) is defined inductively by:

\[
\Phi_d(m) = \Phi_d(m - 1) + \Phi_{d-1}(m - 1)
\]

with initial conditions \(\Phi_d(0) = \Phi_0(m) = 1\).

**Lemma 1**

\[
\Phi_d(m) = \sum_{i=0}^{d} \binom{m}{i}
\]

**Lemma 2**

\[
\sum_{i=0}^{d} \binom{m}{i} \begin{cases} 
2^m & \text{if } m < d \\
(\frac{em}{d})^d & \text{otherwise}
\end{cases}
\]

**Lemma 3**

\[
\Pi_C(m) \leq \Phi_d(m)
\]

The proofs can be found in the textbook (section 3.4). Note that when \(m > d\) this provides a polynomial upper bound on \(\Pi_C(m)\), the number of “behaviors” of \(C\) on a sample of size \(m\). This is the crucial connection to PAC learning.

4 Lower Bounds on Sample Size

**Theorem 1** Any algorithm that PAC learns \(C\) must use at least \(\frac{1}{64} \cdot \frac{d-1}{\epsilon}\) examples, where \(VCD(C) = d\).

**Proof:** The idea is to find a configuration \((\epsilon, \delta, c, D)\) for any PAC learning algorithm such that when the sample size \(m\) satisfies \(m < \frac{1}{64} \cdot \frac{d-1}{\epsilon}\), the output hypothesis will fail the PAC learning conditions. Therefore, in the following context, we assume \(m < \frac{1}{64} \cdot \frac{d-1}{\epsilon}\). We denote \(c\) as a target concept and \(h\) as the output hypothesis output by the respective PAC learning algorithm using \(m\) examples.

Let \(S = \{X_0, X_1, ..., X_{d-1}\}\) be a shattered set (since \(VCD(C) = d\)). Thus, there exist \(2^d\) concepts in \(C\) that produce all possible labelings for \(S\).

Consider the \(2^{d-1}\) concepts in this set that label \(X_0\) as positive and call this set \(\tilde{C}\).

Pick \(\epsilon < \frac{1}{16}\), \(\delta = \frac{1}{16}\), and a discrete probability distribution \(D\) over sample space as:

\[
Prob(x) = \begin{cases} 
0 & x \notin \{X_0, X_1, ..., X_{d-1}\} \\
1 - 16\epsilon & x = X_0 \\
\frac{16\epsilon}{d-1} & x \in \{X_1, X_2, ..., X_{d-1}\}
\end{cases}
\]
Next we consider the question: when the algorithm calls the oracle \( EX(c, D) \) \( m \) times, namely, we obtain \( m \) examples with labels, how many examples in \( \{X_1, X_2, ..., X_{d-1}\} \) does the algorithm see?

We can calculate the expected value using linearity of expectation. We denote \( l \) as the number of examples in \( \{X_1, ..., X_{d-1}\} \) among the \( m \) examples (note that this allows for repetitions) and \( l_i \) as the indicator variable for the \( i \)th example in \( \{X_1, ..., X_{d-1}\} \) for \( i = 1, 2, ..., m \). Thus, we have:

\[
E(l) = \sum_{i=1}^{m} E(l_i) = \sum_{i=1}^{m} 16\epsilon = m \cdot 16\epsilon \leq \frac{1}{64} \cdot \frac{d-1}{\epsilon} \cdot 16\epsilon = \frac{1}{4}(d-1)
\]

By Markov inequality, \( Pr(l > \frac{1}{2}(d - 1)) < \frac{1}{2} \). Finally, because our calculation allows for repetitions, we also have that with probability \( \geq \frac{1}{2} \) we see less than half of the examples in \( \{X_1, X_2, ..., X_{d-1}\} \).

Next we study how the output hypothesis will behave on examples from \( \{X_1, X_2, ..., X_{d-1}\} \). We start by defining the following partial error:

**Definition 2** \( \text{err}'(h) = Pr(x \in \{X_1, ..., X_{d-1}\} \text{ and } h(x) \neq c(x)) \).

Obviously it is bounded by both \( 16\epsilon \) and \( \text{err}(h) \).

Now consider consider running the learning algorithm where we first uniformly sample \( c \in \tilde{C} \) and then obtain samples for \( c \) from distribution \( D \). \( \text{err}'(h) \) is a random variable in \([0, 16\epsilon]\). The expectation of the error can be lower bounded as follows: at least 1/2 of the time we do not see at least 1/2 of the examples. For any of the examples not seen, any prediction has an error of 1/2 when the probability is taken over the choice of \( c \). That is:

\[
E(\text{err}'(h)) > \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 16\epsilon = 2\epsilon
\]

We next use the following claim (to be proved in homework assignment)

**Claim 1** If \( z \) is a random variable in \([0, \beta]\), and \( E(z) > \alpha \), then \( Pr(z \geq \frac{\alpha}{2}) > \frac{\alpha}{2\beta} \).

By setting \( \beta = 16\epsilon \) and \( \alpha = 2\epsilon \), we have:

\[
Pr(\text{err}'(h) \geq \epsilon) > \frac{1}{16}.
\]

This shows that the algorithm fails with probability \( > 1/16 \) in the setting where we choose \( c \) uniformly at random. Therefore \( \exists \tilde{c} \in \tilde{C}, \ s.t. \ Pr(\text{err}'(h) \geq \epsilon) > \frac{1}{16} \) when learning \( \tilde{c} \). Finally because \( \text{err}'(h) \leq \text{err}(h) \) the same holds for \( \text{err}(h) \).

To summarize, when we run the learning algorithm to learn \( \tilde{c} \) using \( m \) examples sampled from \( D \) and obtain the output hypothesis \( h \), we know that \( Pr(\text{err}(h) \geq \epsilon) > \delta \).