a comment on the proof in ch 2 of Understanding Machine Learning

Steven Brawer
11/2017

introduction

This document tries to expand on and rationalize the arguments given in chapter 2 of the book "Understanding Machine Learning" by Shai Shalev-Shwartz and Shai Ben-David. I am by no means an expert statistician (or even a statistician, really), but am proficient enough in engineering mathematics, so this is the attempt by a non-statistician to rationalize a statistical argument.

This is not an exhaustive discussion but only a paraphrase (and expansion) of the proof. It is assumed that the reader has access to the aforementioned book or, equivalently, to the free PDF.

notation

feature - a vector. The dimension does not need to be specified for purposes of this note. A general feature is labeled x_μ, with Greek subscript. All features are distinct, even if they have the same value. So $x_\mu, x_{\mu'}$ are different features even if they are equal in value, $x_\mu = x_{\mu'}$.

X - **domain set** - set of **features**. Domain points are **instances** (a particular feature from the space), X is also called an **instance space.**

Υ - **Label set**, {0, 1}

S - **training data** or **training set**, finite *sequence* of pairs $\mathrm{X} \times \Upsilon$, $\left((x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\right)$

$h : \mathrm{X} \to \Upsilon$ - **prediction rule, hypothesis, predictor, classifier**. It is a function. Output of learner.

$f : \mathrm{X} \to \Upsilon$ - **correct or true labeling function.**

H - hypothesis class - a set of hypotheses $h : \mathrm{X} \to \Upsilon$

Set notation. For example, **{x: C(x)}** - the set of features such that for feature x the condition C(x) is true. It is implied that $x \in \mathrm{X}$ for feature (vector) x. C(x) might be something like $a(x) \neq b(x)$, or $a(x) > \varepsilon$, or a(x)==3 and b(x)<4.

events and probability D[...]

An **event** is a subset of X, so it is either the empty set or a set of one or more features, including X itself (since every set is a subset of itself). Each event (read set) A has an associated probability D[A] of occurring. The probability D is not specified except to assume it is sufficiently well-behaved for our purposes. The probability of the empty set is 0, that is, D[{}]=0.

For example, the probability that some feature has value z (a vector which is in X) is

D[{x: x==z}] (1)

which is the probability associated with the set {x: x==z}. Note that x is a dummy variable, so (1) could well be written

$$D\left[\{hgyd : hgyd == z\}\right] or\ D\left[\{t : t == z\}\right]$$

We might, with some informality, write (1) as

p(z) = D[{x: x==z}] (2)

where p is a function - the probability of the value z (rather than of a set). If there are a finite number of features, and they all have distinct values, the set in (1) will have either 0 or 1 members. Therefore it is generally more useful to look at values in some range, or with some condition that may apply to more than one feature. For example, if a feature is one dimensional,

$$D\left[\{x : a \le x \le b\}\right] \quad (3)$$

can be written as

$$\sum_{\mu} D\left[\left\{z : z == x_{\mu}\right\}\right]\Psi(a \le x_{\mu} \le b)$$

where (my notation) $\Psi = 1$ if x is in [a,b] and 0 otherwise. If the features form a continuum, (3) is

$$D\left[\{x : a \le x \le b\}\right] = \int_{a}^{b} p(x)dx$$

Another example, closer to what will be used later (but assuming X is finite), is the probability that a feature x satisfies the condition G(x)=0:

$$D\left[\{x : G(x) = 0\}\right] = \sum_{\mu} D\left[\left\{z : z = x_{\mu}\right\}\right]\delta_{G(x_{\mu}),0} \quad (4)$$

where

$$\delta_{a,b} \equiv \begin{cases} 1\ if\ a = b \\ 0\ otherwise \end{cases}$$

The **joint probability** of m features is defined as

$$D^{m}[\{x_1, x_2, ..., x_m : G(x_1, x_2, ..., x_m)\}] \quad (5)$$

where G() is some condition on the values. In (5), m is not a power but a superscript. The superscript is just used to indicate a joint probability of a set with m members. All the x_i are dummy variables.

If the m vectors are iid, all with distribution D, and if the condition G can be split

$$G(x_1,x_2,...,x_m) = g_1(x_1)\ and\ g_2(x_2)....and\ g_m(x_m) \tag{6}$$

Then the joint probability DD becomes the product of probabilities

$$D^m[\{x_1,x_2,...,x_m : G(x_1,x_2,...,x_m)\}] = D[\{x1 : g_1(x_1)\}]D[\{x2 : g_2(x_2)\}].... \tag{7}$$

Eq (7) will be crucial later.

true error and training error

The true error of a prediction rule $h : \mathrm{X} \to \Upsilon$ is defined as

$$L_{D,f}(h) \equiv D[\{x : h(x) \neq f(x)\}] \tag{8}$$

The functional $L_{D,f}$ **is a probability,** namely, the probability of choosing an x such that the hypothesis h(x) is incorrect. L is a measure of error. Note that the dummy variable x in (8) is taken over all X. The true error L is independent of any training set or hypothesis class.

There are really two things happening in (8):

1. Choose x according to D.
2. Test whether h(x) = f(x).

The first is a probability, the second is a condition.

So for example, suppose that a particular x0 is highly improbable, but that $h(x0) \neq f(x0)$ for a particular h, then

$$D[\{x : x = x0\ and\ h(x0) \neq f(x0)\}] \ll 1 \tag{9}$$

because it is highly improbable to choose x0, even though h(x0) is not correct. So the error is small because x0 is unlikely, not because of any particular form of h (apart from the fact that $h(x0) \neq f(x0)$).

The error L is not the probability that, *given x,* we have $h(x) \neq f(x)$. This might be a conditional probability on some sort of probabilistic choice of h, given a particular x.

We also have a **training error** for a particular sample S and hypothesis h

$$L_S(h) \equiv \frac{\left|\{i \in [m] : h(x_i) \neq f(x_i)\right|}{m} \qquad where\,[m] = \{1,2,...,m\} \tag{10}$$

where the x are not dummy variables but are in the sample S. We could write the training error as (cf eq (2))

$$L_S(h) \equiv \frac{\sum_{x_i \in S} (1 - \delta_{h(x_i), f(x_i)})}{|\,S\,|} \tag{11}$$

so the only contributions to the sum are those x where $h(x) \neq f(x)$. Note that the training error (10) is a number but not, strictly speaking, a probability. It is independent of D, and only depends on the sample, on h(x) and on f(x).

On the other hand, $L_{D,f}(h)$, eq (8), does depend on D but not on any S.

ERM

ERM stands for **Empirical Risk Minimization**. ERM is a function which takes as argument a hypothesis class H and a sample S, and returns a *single* hypothesis h from H.

First, the quantity

$$\arg\min_H L_S(h) \tag{12}$$

returns a **set** of hypotheses h from H such that each h in the returned set minimizes $L_S(h)$. Note that both S and H are arguments, but the notation indicates that generally this is used with fixed H and variable S. Anyhow, we can write

$$\arg\min_H L_S(h) \subseteq H \tag{13}$$

The quantity

$$ERM_H(S) \tag{14}$$

returns a single function h from the set returned by the *argmin, eq (12)*. It does *not* return a set. Therefore

$$h_S \equiv ERM_H(S) \in \arg\min_H L_S(h) \subseteq H \tag{15}$$

which defines h_S.Note that h_S depends not just on S but on H and f(x) and on whatever magic is going on in ERM to select one h from a possible multitude. Eq (15) is not an algorithm, but rather a specification that an algorithm must satisfy in order to produce the single h.

realizability

A hypothesis class H is **realizable** if there exists an h^* such that

$$L_{D,f}(h^*) = 0 \tag{16}$$

That is, there is in a realizable H a function which is a perfect classifier, and which acts in exactly the same way as f. Since L has as input the entire domain X, we can say that $f(x) = h^*(x)$ for all x in X.

Now consider **any** training set S. Since h^* is assumed to exist in H, then there must exist for every possible sample S an h_S such that $L_S(h_S) = 0$. One possibility is that $h_S = h^*$ (since it is guaranteed that there exists $h^* \in H$) but there may be other hypotheses h for the particular S. If there is more than one, we don't know which one ERM will return, but it will return some hypothesis with 0 training error. The particular one is notated h_S.

generating samples

It is useful to have a shorthand notation involving the probability of entire samples. Let $S|_x$ be a **sample instance** (see *notation* section). Consider the probability

$$D^m\left[\left\{S|_x : G\left(S|_x\right)\right\}\right] \tag{17}$$

where G is some condition operating on the features in a sample. Eq (17) is really shorthand for

$$D^m[\left\{x_1, x_2, ..., x_m : S|_x \equiv \left\{x_1, x_2, ..., x_m\right\} \; and \; G\left(S|_x\right)\right\} \tag{18}$$

(Note that the condition $a \equiv b$ is a definition and is always true.) If G is separable, as in eq (6), then the above is equivalent to eq (7).

Eq (17), then, is the probability of finding samples such that the condition G is true for those samples.

the problem - bounding the true error

The problem is to bound eq (20) below.

We assume that H is finite and realizable.

We let $S|_x$ be an sample instance (see previous section). To quote the book: "*The samples in the training set are independently and identically distributed (iid) according to distribution D.*" For given instance, we determine h_S from eq (15), so that $L_S(h_S) = 0$ as discussed above. So there is 0 training error for h_S. On the other hand, the true error $L_{D,f}(h_S)$ is generally not zero - that is, in general

$$L_{D,f}(h_S) > 0 \tag{19}$$

So the problem is this. For any training set, in general there will exist at least one hypothesis which has 0 training error but non-zero true error. How can we put a bound on the true error? That is, can we specify ε, δ such that $0 < \varepsilon < 1, 0 < \delta < 1$

such that

$$D^m\left[\left\{S|_x : L_{D,f}(h_S) > \varepsilon\right\}\right] < \delta \tag{20}$$

The LHS of eq (20) is the probability of choosing a training set $S|_x$ such that that the true error of a particular hypothesis function h_S of that training set is bigger than some value ε. (Note the possibly *confusing notation in (20),* where S stands for $S|_x$.) As discussed between eqs (8) and (9), the LHS of eq (20) is the **probability of choosing** a training set with a particular condition which is being bounded. We are *not* bounding the probability of choosing a particular hypothesis h *given* a training set.

If $\varepsilon = 0$, then the only acceptable hypothesis h is one where the true error is 0. As ε becomes larger than 0, larger true errors become more and more acceptable (depending on δ), and this means the number of training sets producing those errors will be expected to increase.

The idea of eq (20) is that the probability of choosing a bad training set should be small, where a bad training set is one with 0 training error but unacceptable true error. The fact that H is assumed realizable means that the hypothesis being tested has 0 training error. The measure of how bad is unacceptable is given by the combination ε, δ.

Note that δ is a function of m and ε (for fixed D, H, f). For a given δ, as ε decreases, we expect that m must increase in order to maintain the function $\delta(m, \varepsilon)$ constant, so larger and larger samples will be necessary for more accurate hypotheses (those with smaller true error).

The problem is to find the bound δ, eq (20) (actually, to show that a bound exists.) A solution (with what is expected to be a very very large but finite bound) is given in the next section.

the solution

Define two sets, A and M. (The set A is the one used in eq (20).)

$$A \equiv \left\{S|_x : L_{D,f}(h_S) > \varepsilon\right\} \tag{21}$$

$$M = \bigcup_{h \in H} \left\{ S|_x : L_{D,f}(h) > \varepsilon \; and \; L_S(h) = 0 \right\} \tag{22}$$

where the set A involves just one particular h, while M is a union over all relevant h. Note that the LHS of eq (20) is $D^m[A]$. Each of A, M is a *set of sample sets* S.

Note the use of h_S in (21) but of h in (22). So (22) makes use of all h returned by (12) while (21) just uses ERM, eq (14). Certainly the set M includes the set A because M is A with a possibly larger number of hypotheses. (It would be redundant to add $L_S(h_S) = 0$ to set A.) Therefore we have

$$A \subseteq M \tag{23}$$

and therefore, from basic probability theory,

$$D^m[A] \leq D^m[M] \tag{24}$$

So if we find a bound for $D^m[M]$, it is also a bound in eq (20). **The goal now is to find a bound for $D^m[M]$.**

Since H is finite, the union, eq (22), is finite. For any finite (or countable) union we have

$$D\left[\bigcup_n W_n\right] \leq \sum_n D[W_n] \tag{25}$$

for sets W_n. The equality holds only if the sets W are all mutually exclusive - ie, non-intersecting.Therefore, combining (22) and (25),

$$D^m[M] \leq \sum_{h \in H} D^m\left[\left\{ S|_x : L_{D,f}(h) > \varepsilon \; and \; L_S(h) = 0 \right\}\right] \tag{26}$$

Eq (26) is a bound on eq (20), but not a suitable bound. The LHS of (26) is less than 1, since it is a probability. On the other hand, the RHS of eq (26) can be greater than 1.

Each probability in the sum of (26) is the **probability of a set of training sets S.** For two different h, the two corresponding sets of S's are not necessarily mutually exclusive. That is, in general we expect that the sets of selected training sets S may have a non-null intersection. The sum of probabilities of non-mutually-exclusive events (ie, of intersecting sets) may be greater than 1. So eq (26) by itself cannot be taken as a suitable bound for eq (20), since the bound δ must be less than 1. Therefore, we must proceed farther, to ensure that the RHS of eq (26) is less than 1.

In eq (26), $L_{D,f}(h)$ is independent of $S|_x$ (cf eq (8)). So if, for some h = h0, we find that $L_{D,f}(h0) \le \varepsilon$, then $D^m\left[\left\{S|_x : L_{D,f}(h0) > \varepsilon \; and \; L_S(h) = 0\right\}\right] = 0$, so that h0 is not included in the sum on the RHS of eq (26). Therefore we can write

$$D^m[M] \le \sum_{h \in H} \theta\left(L_{D,f}(h) - \varepsilon\right) D^m\left[\left\{S|_x : L_S(h) = 0\right\}\right] \tag{27}$$

where

$$\theta(x) \equiv \begin{Bmatrix} 1 & if \; x > 0 \\ 0 & otherwise \end{Bmatrix}$$

is the step function.

We now make use of the iid assumption to break up the probability D^m , eq (26), into the product of probabilities of individual features, along the lines of eq (7). Note that, for any h, the condition $L_S(h) = 0$, eq (10), can be written as the condition

$$(L_S(h) = 0) = h(x_1) = f(x_1) \; and \; h(x_2) = f(x_2) \; and \ldots and \; h(x_m) = f(x_m) \tag{28}$$

where all x_i are in $S|_x$ .Therefore, eq (7) applies to each term in the sum (27). Also note that each x_i is a dummy variable, and because (17) and (18) are equivalent,

$$D^m[M] \le \sum_{h \in H} \theta\left(L_{D,f}(h) - \varepsilon\right)\left(D\left[\left\{x : f(x) = h(x)\right\}\right]\right)^m \tag{29}$$

It should be obvious that

$$D\left[\left\{x : f(x) = h(x)\right\}\right] + D\left[\left\{x : f(x) \ne h(x)\right\}\right] = 1 \tag{30}$$

The second term on theLHS of eq (30) is just $L_{D,f}(h)$ (eq (8)), so that

$$D\left[\left\{x : f(x) = h(x)\right\}\right] = 1 - L_{D,f}(h) \tag{31}$$

Therefore, using eq (31), eq (29) becomes

$$D^m[M] \le \sum_{h \in H} \theta\left(L_{D,f}(h) - \varepsilon\right)\left(1 - L_{D,f}(h)\right)^m \tag{32}$$

If $L > \varepsilon$, then $1 - L < 1 - \varepsilon$, and letting |H| = size of H,

$$D^m[M] \le \sum_{h \in H} \theta\left(L_{D,f}(h) - \varepsilon\right)(1-\varepsilon)^m \le |H|(1-\varepsilon)^m \tag{33}$$

and therefore, from eq (20), we can choose

$$\delta = |H|(1-\varepsilon)^m \tag{34}$$

Since $1-\varepsilon < 1$, if m is large enough, then $\delta < 1$ as well, and the bound in eq (33) is legitimate.

Suppose, however, that the total number of features available is too small to make $\delta < 1$ based on eq (34). Then, choosing m = total number of features, there is only one training set S and it includes all the features. Since the hypothesis class is assumed realizable, there exists an h* in H such that the true error must be 0 - that is, $L_{D,f}(h^*) = L_S(h^*) = 0$ - and the above argument is not needed - there are no probabilities.

Assuming now that m can be sufficiently large that (34) represents a bound, and assuming that $|H| > \delta$, we have

$$m \ge -\ln\frac{|H|}{\delta}\frac{1}{\ln(1-\varepsilon)} \tag{35}$$

the greater than or equal sign because m must be an integer. Note that $\ln(1-\varepsilon) < 0$. Expanding the logarithm in a Taylor series, with $\varepsilon > 0$,

$$\ln(1-\varepsilon) = -\sum_{j=1}\frac{1}{j}\varepsilon^j < -\varepsilon$$

we see finally that

$$m \ge \ln\frac{|H|}{\delta}\frac{1}{\varepsilon} \tag{36}$$

Recall that m is the number of features in a training set such that eq (20) is valid for given ε, δ. As discussed between eqs (20) and (21), we see that, as ε decreases, for *constant* δ and |H|, the number of features m in the training set must increase.

discussion

Briefly, besides the formulation of the problem as eq (20), the key observations in the proof are eqs (24) and (25), and especially (24). These observations require some intuition, while the remaining steps are straightforward substitution.

Eq (20) and (35) are the desired results, along with the implicit observation that m is greater than the total number of available features. If not, then m should equal the total number of features, and we can set $\varepsilon = \delta = 0$ in eq (20).