PAC Learning and the VC Dimension

prof. dr Arno Siebes

Algorithmic Data Analysis Group
Department of Information and Computing Sciences
Universiteit Utrecht

PAC Learning

Recall the general definition of PAC learning:

A hypothesis class $\mathcal{H}$ is agnostic PAC learnable with respect to a set Z and a loss function $l : Z \times \mathcal{H} \to \mathbb{R}_+$ if there exists a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm A with the following property:

- for every $\epsilon, \delta \in (0,1)$
- for every distribution $\mathcal{D}$ over Z
- when running A on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. samples generated by $\mathcal{D}$
- A returns a hypothesis $h \in \mathcal{H}$ such that with probability at least $1 - \delta$
$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon$$

So What?

The definition of PAC learning tells us

- when we consider we can learn something

It tells us precious little

- about *what* we can learn and *how* we learn that.

This is, of course, to be expected

- you have to know what you want before you can (try to) achieve it

So, this is what we turn to next:

- discover what can be PAC learned and how

ϵ-Representative

How well you can learn an hypothesis from a sample obviously depends on the quality of that sample

- you can't learn anything from a bad sample, it'll tell you that a bad hypothesis is good and a good one bad.

When is a sample good? Simple:

- a sample is good if the estimated quality (i.e., the loss) of an hypothesis on that sample is close to its true loss (under the distribution)

More formally:

A data set D is called ϵ-representative wrt domain Z, hypothesis class $\mathcal{H}$, loss function l and distribution $\mathcal{D}$ if

$$\forall h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| \leq \epsilon$$

ϵ-Representative Samples are Good

Lemma

Let data set D be $\epsilon/2$-representative wrt domain Z, hypothesis class $\mathcal{H}$, loss function l and distribution $\mathcal{D}$. Then any output of $ERM_{\mathcal{H}}(D)$ – i.e., any $h_D \in \operatorname*{argmin}_{h\in\mathcal{H}} L_D(h)$ – satisfies

$$L_{\mathcal{D}}(h_D) \leq \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$$

Proof

for any $h \in \mathcal{H}$:

$$L_{\mathcal{D}}(h_D) \leq L_D(h_D)+\epsilon/2 \leq L_D(h)+\epsilon/2 \leq L_{\mathcal{D}}(h)+\epsilon/2+\epsilon/2 = L_{\mathcal{D}}(h)+\epsilon$$

That is, on an $\epsilon/2$-representative sample D, the $ERM_{\mathcal{H}}$-rule yields an optimal result

- optimal in the sense: as good as it gets.

Uniform Convergence

If ϵ-representative samples allow us to learn as good as possible, we can PAC learn if we can guarantee that we will almost always get ϵ-representative samples:

A hypothesis class $\mathcal{H}$ has the *uniform convergence property* wrt domain Z and loss function l if

- there exists a function $m_{\mathcal{H}}^{UC} : (0,1)^2 \to \mathbb{N}$
- such that for all $(\epsilon, \delta) \in (0,1)^2$
- and for any probability distribution $\mathcal{D}$ on Z

If D is an i.i.d. sample according to $\mathcal{D}$ over Z of size $m \geq m_{\mathcal{H}}^{UC}(\epsilon, \delta)$. Then D is ϵ-representative with probability of at least $1 - \delta$.

A Tool to Prove PAC

The preceding discussion gives a tool to prove that we can PAC learn a hypothesis class: just prove that it has the uniform convergence property. More formally, we have

Corollary

If hypothesis class $\mathcal{H}$ has the uniform convergence property with function $m_{\mathcal{H}}^{UC}$, then $\mathcal{H}$ is agnostically PAC learnable with sample complexity $m_{\mathcal{H}}(\epsilon, \delta) \leq m_{\mathcal{H}}^{UC}(\epsilon/2, \delta)$. Moreover, $ERM_{\mathcal{H}}$-rule is successful agnostic PAC learner for $\mathcal{H}$.

Using this tool, we'll now prove that finite hypothesis classes are agnostically PAC learnable.

What We Need to Prove

To prove that finite hypothesis classes have the uniform convergence property, we need to

- find for fixed ϵ and δ
- a sample size m
- such that for any distribution D
- an i.i.d. sample D of size $|D| \geq m$
- with probability at least $1 - \delta$

$$\forall h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| \leq \epsilon$$

That is, we need to prove that

$$\mathcal{D}^m \left(\{ D \mid \forall h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| \leq \epsilon \} \right) \geq 1 - \delta$$

Which is equivalent to

$$\mathcal{D}^m \left(\{ D \mid \exists h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon \} \right) < \delta$$

Some Simple Algebra

Clearly,

$$\{D \mid \exists h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\} = \bigcup_{h \in \mathcal{H}} \{D \mid |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}$$

we first gather per h the "bad" samples, and then take the union of that set.

Probability theory (the union bound) then tells us that

$$\mathcal{D}^m\left(\{D \mid \exists h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}\right) \\ \leq \sum_{h \in \mathcal{H}} \mathcal{D}^m\left(\{D : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}\right)$$

So if we can prove that for a large enough m each of the terms

$$\mathcal{D}^m\left(\{D : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}\right)$$

is small enough, we are done.

Expectations

Recall that

- $L_{\mathcal{D}}(h) = E_{d \sim \mathcal{D}}\left(l(d,h)\right)$
- while L_D is the expected value (i.e., the average) of the loss on the sample D, i.e., $L_D(h) = \frac{1}{m}\sum_{i=1}^{m} l(d_i,h)$

Each $l(d_i,h)$ is itself a random variable (because d_i is randomly sampled), its expected value is

- $L_{\mathcal{D}}(h)$

Since expectation is a linear operator ($\mathbb{E}(A+B) = \mathbb{E}(A) + \mathbb{E}(B)$)

- $\mathbb{E}(L_D(h)) = \mathbb{E}\left(\frac{1}{m}\sum_{i=1}^{m} l(d_i,h)\right) = L_{\mathcal{D}}(h)$

This means that we aim to prove that the expected loss on a (large enough) sample is close to the true expected loss

- but that is exactly what the concentration inequalities we discussed in the second session measure

We are going to use Hoeffding's inequality

Hoeffding

Recall Hoeffding:
Let $Z_1, \ldots, Z_m$ be a sequence of i.i.d. random variables and let $\bar{Z} = \frac{1}{m}\sum_{i=1}^m Z_i$. Furthermore, Let $\mathbb{E}(\bar{Z}) = \mu$ and assume that $\mathbb{P}[a \leq Z_i \leq b] = 1$, for every i. Then, for any $\epsilon > 0$, we have

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^m Z_i - \mu\right| > \epsilon\right] \leq 2e^{-\left(\frac{2m\epsilon^2}{(b-a)^2}\right)}$$

to use this inequality, for a fixed h

- let the $l(d_i, h)$ be the random variables θ_i
- and assume that the loss is bounded, i.e., $l(d_i, h) \in [a, b]$

Then we have that

- $L_D(h) = \frac{1}{m}\sum_{i=1}^m l(d_i, h) = \frac{1}{m}\sum_{i=1}^m \theta_i$
- $\mathbb{E}(L_D(h)) = L_{\mathcal{D}}(h) = \mu$

Now we can substitute

Substitute and Compute

Substitution gives us

$$\begin{aligned}\mathcal{D}^m\left(\{D : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}\right) &= \mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i - \mu\right| > \epsilon\right] \\ &\leq 2e^{-\left(\frac{2m\epsilon^2}{(b-a)^2}\right)}\end{aligned}$$

Hence we have that

$$\begin{aligned}\mathcal{D}^m\left(\{D \mid \exists h \in \mathcal{H} : |L_D(h) - L_{\mathcal{D}}(h)| > \epsilon\}\right) &\leq \sum_{h\in\mathcal{H}} 2e^{-\left(\frac{2m\epsilon^2}{(b-a)^2}\right)} \\ &= 2|\mathcal{H}|e^{-\left(\frac{2m\epsilon^2}{(b-a)^2}\right)}\end{aligned}$$

Solving $\delta = 2|\mathcal{H}|e^{-\left(\frac{2m\epsilon^2}{(b-a)^2}\right)}$ for m gives us the desired result.

Finite Hypothesis Classes are PAC

Let $\mathcal{H}$ be a finite hypothesis class, let Z be a domain, and let $l : Z \to [a, b]$ be a (bounded) loss function. Then $\mathcal{H}$ has the uniform convergence property with sample complexity

$$m_{\mathcal{H}}^{UC}(\epsilon, \delta) \leq \left\lceil \frac{(b-a)^2 \log(2|\mathcal{H}|/\delta)}{2\epsilon^2} \right\rceil$$

Moreover, $\mathcal{H}$ is agnostically PAC learnable using the ERM rule with sample complexity

$$m_{\mathcal{H}}(\epsilon, \delta) \leq m_{\mathcal{H}}^{UC}(\epsilon/2, \delta) \leq \left\lceil \frac{2(b-a)^2 \log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil$$

Discussing this Result

By going from realizability to agnostic and from $0/1$ loss to a general loss function, we go

- from $m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \right\rceil$
- to $m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{2(b-a)^2 \log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil$

The biggest difference is

- that the denominator goes from ϵ to ϵ^2

Which means that for the same level of accuracy

- the minimal sample size grows by a factor of $1/\epsilon$

The same as we had already seen for inconsistent learning

- which is not surprising since we use the Hoeffding inequality in both cases.

The contribution of a general loss functions is smaller

- and can often be normalized to $[0, 1]$

Finite Reconsidered

When we first restricted ourselves to the finite case, we already remarked

- that this is not a strong limitation, it contains already many practical examples, such as using 256 bit reals for thresholds, or all Python programs of at most 10^{32} characters

If we take the former

- we have 2^{256} different hypothesis

Which because of the log

- gives only a "factor" of 256 in the sample size

And given that we have a (per step) efficient algorithm to learn thresholds

- we can learn "unrestricted" thresholds efficiently in practice

The same is unfortunately not known for the Python programs.

A Note of Caution

The fact that $\mathcal{H}$ is agnostically PAC learnable using the ERM rule

- doesn't mean that the result is any good

It only means that you can be reasonably sure that

- the ERM rule gives you a result that is close to the optimal result.

If the optimal result is bad

- because, e.g., the chosen hypothesis class fits the data really badly

The ERM rule will also give you a bad result.

PAC doesn't tell you that your hypothesis class fits the data well

- it only tells you that *if* it fits well, the ERM rule will probably give you a reasonable good hypothesis.

Bias and Variance

We can decompose our error as:

$$L_{\mathcal{D}}(h_D) = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon_{est}$$

The first term is the approximation error

- it measures how well our hypothesis class fits the distribution
- it is independent of the particular sample
- it is the *bias* term

The second term is the estimation error

- it measures how well our particular sample let us estimate the best classifier
- it *varies* with samples
- it is the *variance* term

Isn't Bias Bad?

Having en estimation error is inevitable if you work with samples

- and working with samples is itself inevitable

But a bias term? Bias is bad isn't it?

- bias is prejudice!

This is, both true and not true

- true in the sense that we should aim to minimize the effect of our bias
  - we want to minimize $L_{\mathcal{D}}(h_D)$
  - and minimizing $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$, by choosing an appropriate $\mathcal{H}$, is an important component of this
- not true, because it represents our background knowledge
  - if we understand our data generating process well
  - we can choose $\mathcal{H}$ such that $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) = 0$
  - i.e., such that the realizability assumption holds
  - if we don't understand it well enough, we'll make mistakes

We cannot learn perfectly without the proper background knowledge

No Free Lunch

Let A be any learning algorithm for a binary classifier wrt $0/1$ loss over domain X. Let m be any number smaller than $|X|/2$. Then there exists a distribution $\mathcal{D}$ over $X \times \{0, 1\}$ such that

- there exists a function $f : X \to \{0, 1\}$ with $L_{\mathcal{D}}(f) = 0$
- with probability of at least $1/7$ over the choice of $D \sim \mathcal{D}^m$ we have that $L_{\mathcal{D}}(A(D)) \geq 1/8$

In other words, for every learning algorithm there are (in a sense pathological) cases

- cases for which this algorithm will fail miserably

It is not this special case that is the problem

- another algorithm A' may do completely fine

it simply means that an adversary can use the fact that A has no clue what happens on the other half of the domain. We cannot learn perfectly without the proper background knowledge

Setting Up the Proof

To prove the theorem, a distribution is constructed that will stymie A. The basic ingredients are:

- Let $C \subseteq X$, such that $|C| = 2m$
- There exist $T = 2^{2m}$ functions from C to $\{0, 1\}$
- Denote these functions by $f_1, \ldots, f_T$.
- For each function f_i define the distribution $\mathcal{D}_i$ by

  $$\mathcal{D}_i = \begin{cases} 1/|C| & \text{if } y = f_i(x) \\ 0 & \text{otherwise} \end{cases}$$

  That is, $\mathcal{D}_i$ is perfect for f_i. it will only generate samples on which f_i is correct ($y = f_i(x)$)
- Hence $L_{\mathcal{D}_i}(f_i) = 0$

Proof

Claim: for every learning algorithm A that receives a sample D of m elements of $C \times \{0,1\}$ and returns a function $A(D) : C \to \{0,1\}$:

$$\max_{i \in [T]} E_{D \sim \mathcal{D}_i^m} \left[L_{\mathcal{D}_i}(A(D)) \right] \geq 1/4$$

Assuming the claim, we have (by choosing a maximizing i) that for every learning algorithm A' that receives m examples from $X \times \{0,1\}$ there exists a function $f : X \to \{0,1\}$ and a distribution $\mathcal{D}$ over $X \times \{0,1\}$, such that $L_{\mathcal{D}}(f) = 0$ and

$$E_{D \sim \mathcal{D}^m} \left[L_{\mathcal{D}}(A'(D)) \right] \geq 1/4$$

Wlog assuming that our loss bounded to $[0,1]$, Markov's inequality tells us that for the random variable $L_{\mathcal{D}}(A'(D))$

$$\mathbb{P} \left(L_{\mathcal{D}}(A'(D)) \geq 1/8 \right) \geq 1/7$$

The Intuition Behind the Claim

There are $k = (2m)^m$ possible samples of m elements from C.

- which is but a fraction of the 2^{2m} distributions (and functions) on C we just created

That is, there for each such sample of m elements

- there are many distributions that are consistent with it

Given their sheer number

- it is not surprising that for each such distribution there is another one that is very different on the m *unseen* elements of C

Since each algorithm chooses 1 of these possible "continuations"

- it has to fail against a cunning adversary.

All possible functions is too rich a class to learn

- if you can predict everything, you can predict nothing

Corollary

A corollary of the No Free Lunch Theorem is:

> *Let X be an infinite domain and let $\mathcal{H}$ be the set of all functions from X to $\{0, 1\}$. The $\mathcal{H}$ is not PAC learnable.*

In the end, for the simple reason that $|X| > 2m$ for every m, although it is slightly more subtle.

So, we cannot circumvent the bias term by taking a very rich class of hypotheses.

- in fact by enlarging $|\mathcal{H}|$ we may make the bias term smaller, but we will make the estimation term larger because of the $\log |\mathcal{H}|$ in that term.

The Bias/Variance trade off is a inherent aspect in learning

- it is a manifestation of the problem of induction

Is This It?

So, we know that

- finite cases can be PAC learned
    - but mind the bias
- and that a very rich infinite class cannot

Is there anything in between?

- an infinite class that can be PAC learned?

Fortunately, there are such classes. We first look at a concrete one

- thresholds again, of course

And then we start working on the general case

- by defining the VC dimension

In a week we will see that PAC learning is characterized by a finite VC dimension.

Threshold Learning Reconsidered

We have seen that under the realizability assumption, threshold functions cannot be PAC learned.

- in the end because there are (uncountable) infinitely many options

By losing that assumption, we only have to get close to the true value

- hence, all we have to prove is that whatever distribution there is, the ERM rule will most probably get us close

Recall that in this case the ERM rule maintains an interval in which the true value lies

- we know that all values to the left are classified as negative, while all values to the right are classified as positive

So, let a^* be the true vale and define $a_1, a_2 \in \mathbb{R}$ such that

$$\mathbb{P}_{x \sim D}(x \in (a_1, a^*)) = \mathbb{P}_{x \sim D}(x \in (a^*, a_2)) = \epsilon$$

If we now can prove that we most likely get an example from this interval, we are done

Computing the Bound

Let b_1 be the largest element in data set D with $b_1 \leq a^*$ and let b_2 be the smallest element of d with $a^* \leq b_2$. Then

$$\mathbb{P}_{D\sim\mathcal{D}^m}\left[L_{\mathcal{D}}(h_D) > \epsilon\right] \leq \mathbb{P}_{D\sim\mathcal{D}^m}\left[b_1 < a_1\right] + \mathbb{P}_{D\sim\mathcal{D}^m}\left[b_2 > a_2\right]$$

Now, note that

$$\begin{aligned}\mathbb{P}_{D\sim\mathcal{D}^m}\left[b_1 < a_1\right] &= \mathbb{P}_{D\sim\mathcal{D}^m}\left[\forall x \in D : x \notin (a_1, a^*)\right] \\ &= (1-\epsilon)^m \leq e^{-\epsilon m}\end{aligned}$$

and similarly, $\mathbb{P}_{D\sim\mathcal{D}^m}\left[b_2 > a_2\right] \leq e^{-\epsilon m}$.
Hence, with a sample complexity

$$m_{\mathcal{H}}(\epsilon, \delta) \leq \lceil \log(2/\delta)/\epsilon \rceil$$

the derivation we gave shows the PAC learnability of threshold functions.

Free Lunches vs Thresholds

So, why are threshold classifiers not a victim of the no free lunch theorem?

- after all, we can PAC learn them

The reason is simple

- the class of threshold classifiers is so simple that an adversary has no room to create an adversarial distribution

In fact, as our discussion above shows

- if two threshold classifiers agree on a large enough sample
- their respective thresholds will be close to each other
- there is no way you can force them to behave completely differently on unseen examples.

If that would have been possible,

- we would have been able to create an adversarial distribution.

So, it seems necessary for PAC learnability that $\mathcal{H}$ isn't too expressive

How Expressive is $\mathcal{H}$?

In our classification context, a hypothesis is simply a function:

$$h : X \rightarrow \{0, 1\}$$

Hence the expressiveness of $\mathcal{H}$

- is necessarily a measure of how many functions $\mathcal{H}$ can express.

In the light of the No Free Lunch theorem, not only functions on X, but also functions on (finite) subsets of X

Let $\mathcal{H}$ be a set hypotheses, i.e., of functions from X to $\{0, 1\}$, and let C be a (finite) subset of X. The *restriction* of $\mathcal{H}$ to C, denoted by $\mathcal{H}_C$, is the set of functions from C to $\{0, 1\}$ that can be derived from $\mathcal{H}$.

C is *shattered* by $\mathcal{H}$ if $\mathcal{H}_C$ is the set of all functions from C to $\{0, 1\}$, i.e., if

$$|\mathcal{H}_C| = 2^{|C|}$$

Vectors for Ease of Notation

Because C is a finite set, we can enumerate its elements

- $C = \{c_1, c_2, \ldots, c_n\}$

Restricting $h \in \mathcal{H}$ to C is simply

- $h_C = \{h(c_1), h(c_2), \ldots, h(c_n)\}$

If we fix an order on the elements of C

- which we can do easily because it is finite

The restriction becomes a vector over $\{0, 1\}^{|C|}$

- $h_C = (h(c_1), h(c_2), \ldots, h(c_n))$

And $\mathcal{H}$ shatters C exactly when

- $\{h_C \mid h \in \mathcal{H}\}$ equals the set of all vectors in $\{0, 1\}^{|C|}$
- i.e., if we can construct every $0/1$ vector of length $|C|$

Thresholds Again

Let C be a one element set $C = \{c_1\}$

- e.g., $c_1 = \pi$

There are two $0/1$ vectors of length 1

- (0) and (1)

And the thresholds 3 and 4

- map π to 1 and 0 respectively

Hence, $C = \{c\}$ is shattered by $\mathcal{H}$

For $C = \{c_1, c_2\}$, wlog $c_1 < c_2$

- say $c_1 = 0$ and $c_2 = 1$,

however, we can create only three vectors

- $(0, 0)$, $(0, 1)$, and $(1, 1)$

It is impossible to create $(1, 0)$

- there is no θ such that $\theta \leq c_1 \wedge \theta \geq c_2$ since $c_1 < c_2$

Hence $C = \{c_1, c_2\}$ is *not* shattered by $\mathcal{H}$

Free Lunches Revisited

If you recall the Proof of the No Free Lunch theorem, you'll see that we can create the same adversarial distribution if $\mathcal{H}$ shatters a too large class.

Let $\mathcal{H}$ be a hypothesis class of functions $h : X \to \{0, 1\}$ and m a training set size. If there exists a set $C \subseteq X$ of size $2m$ that is shattered by $\mathcal{H}$, then for any learning algorithms A there exists a distribution $\mathcal{D}$ over $X \times \{0, 1\}$ such that

- there exists a function $f : X \to \{0, 1\}$ with $L_{\mathcal{D}}(f) = 0$
- with probability of at least $1/7$ over the choice of $D \sim \mathcal{D}^m$ we have that $L_{\mathcal{D}}(A(D)) \geq 1/8$

Shattering is good, but don't shatter too much.

The VC Dimension

The VC dimension of a set of hypotheses $\mathcal{H}$ is the size of the largest set $C \subseteq X$ such that C is shattered by $\mathcal{H}$. If $\mathcal{H}$ can shatter arbitrary sized sets, its VC dimension is infinite.

VC comes from Vapnik and Chervonenkis whom we encountered before.

A simple consequence of the previous slide is:
Let $\mathcal{H}$ be a hypothesis class of infinite VC dimension, then it is *not* PAC learnable

VC Dimension Example

Our discussion of shattering by threshold functions shows that

- this set has VC dimension 1

Now consider the set of interval predictors

- $h_{[a,b]}(x) = 1$ iff $x \in [a, b]$

Let $C = \{0, 1\}$ then

- $h_{[-2,-1]} = (0, 0)$
- $h_{[-2,0]} = (1, 0)$
- $h_{[1,2]} = (0, 1)$
- $h_{[-1,2]} = (1, 1)$

Hence the VC dimension is ≥ 2. Now, note that for $C = \{c_1, c_2, c_3\}$ with $c_1 \leq c_2 \leq c_3$ we can not create $(1, 0, 1)$; if c_1 and c_3 are in the interval, so is c_2. In other words,

- the VC dimension of interval classifiers is 2.

Linear Algebra

You will probably remember from high school that a line in the plane is given by

$$y = ax + b$$

For vectors

$$\vec{a} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}, \vec{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^n$$

their *dot product* is defined by

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{n} a_1 b_i$$

So, if we translate the traditional (x, y) in the plane to the standard vector notation $(x_1, x_2)^T$ we specify a line by

$$\vec{w} \cdot \vec{x} + \vec{b} = 0$$

Hyperplanes as Classifiers

If we take the vectors $\vec{x}, \vec{w}, \vec{b} \in \mathbb{R}^n$. The equation

$$\vec{w} \cdot \vec{x} + \vec{b} = 0$$

specifies a *hyperplane* in $\mathbb{R}^n$

Such a hyperplane can be used as a (simple) classifier

- points above the plane belong to one class
- points below the plane to the other class

In other words. the classifier is given by

$$f_{\left(\vec{w},\vec{b}\right)}\left(\vec{x}\right) = sign\left(\vec{w} \cdot \vec{x} + \vec{b}\right)$$

What would the VC dimension of this class be?. Let us start with lines in the plane

Three Points in the Plane

If we have $c_1, c_2, c_3 \in \mathbb{R}^2$ *not* on one line, it is easy to with a line classifier we can construct $(-1, -1, -1)$ and $(1, 1, 1)$ easily

- draw the line above (below) all three points

All other cases give one point a different class then the other 2; say c_1 and c_2 are labelled as $+1$ and c_3 by -1

- draw the line l_1 through c_1 and c_2
- draw the perpendicular from c_3 to l_1 and determine p_4 halfway c_3 and l_1
- draw l_2 parallel to l_1, through p_4

Clearly l_2 separates the two classes. Convince yourself thst this means that there is a set of three points in $\mathbb{R}^2$ that is shattered by a line

Hence the VC dimension is ≥ 3

Four Points in the Plane

Let $c_1, c_2, c_3, c_4 \in \mathbb{R}^2$

- if three or more are on one line, they can not be shattered by a line

If no three are on one line, we can draw

- six lines that connect the different points

Clearly

- two of these lines, say l_1 and l_2, cross

If we now label

- the c_i on l_1 by $+1$
- and the c_i on l_2 by -1

there is *no* line that separates the two classes

Hence the VC dimension of lines in the plane is 3.

The VC dimension of Hyperplane Classifiers

Exercise: Prove the the VC dimension of hyperplane classifiers in $\mathbb{R}^n$ is $n+1$.