

Lecture 9 - 07-04-2020

\hat{h} is ERM predictor

$$\ell_D(\hat{h}) \leq \min \ell_D(h) + \sqrt{\frac{2}{m} \ln \frac{2H}{\delta}} \quad \text{with prob. at least } 1 - \delta$$

Now we do it with tree predictors

1.1 Tree predictors

$$X = \{0, 1\}^d \longrightarrow \text{Binary classification}$$

$$h : \{0, 1\}^d \longrightarrow \text{Binary classification } H1$$

How big is this class?

Take the size of codomain power the domain $\longrightarrow |H| = 2^{2^d}$

Can we have a tree predictor that predict every H in this class?

For every $h : \{0, 1\}^d \longleftarrow \{-1, 1\} \quad \exists T$

We can **build a tree** such that $h_T = h$

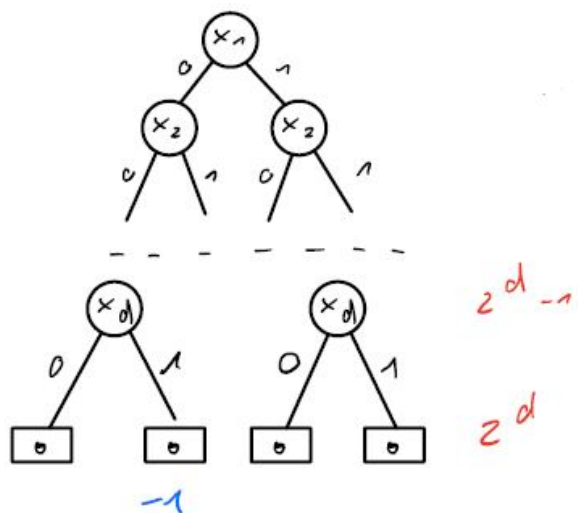


Figure 1.1: Tree building

$$X = (0, 0, 1, \dots, 1) \quad h(x) = -1$$

$x_1, x_2, x_3, \dots, x_d$

I can apply my analysis to this predictors

If I run ERM on H

$$\ell_D(\hat{h}) \leq \min \ell_D(\hat{h}) + \sqrt{\frac{2}{m} 2^d \ln 2 + \ln \frac{2}{\delta}} \quad \rightarrow \ln |H| + \ln \frac{2}{\delta}$$

No sense! What we find about training set that we need?

Worst case of overfitting $m \gg 2^D = |X| \Rightarrow$ training sample larger

PROBLEM: cannot learn from a class too big (H is too big)

I can control H just limiting the number of nodes.

$H_N \rightarrow$ tree T with at most N nodes, $N \ll 2^D$

$|H_N| = ?$

$|H_N| = (\# \text{ of trees with } \leq N \text{ nodes}) \times (\# \text{ of test on internal nodes}) \times (\# \text{ labels on leaves})$

$$|H_N| = \otimes \times d^M \times 2^{N-M}$$

N of which $N - M$ are leaves

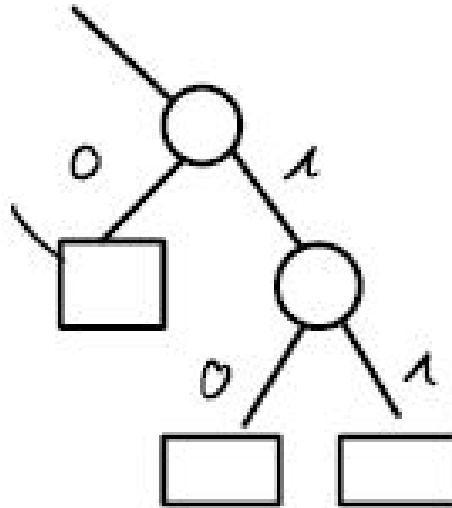


Figure 1.2: Tree with at most N nodes

\otimes # of binary trees with N nodes, called *Catalan Number*

1.1.1 Catalan Number

*We are using a binomial *

$$\frac{1}{N} \binom{2N-2}{N-1} \leq \frac{1}{N} \left(e \frac{(2N-2)}{N-1} \right)^{N-1} = \frac{1}{N} (2e)^{N-1}$$

$$\binom{N}{K} \leq \left(\frac{en}{k} \right)^k \quad \text{from Stirling approximation}$$

Counting the number of tree structure: a binary tree with exactly N nodes. Catalan counts this number. \rightarrow **but we need a quantity to interpret easily**. So we compute it in another way.

Now we can rearrange everything.

$$|H_N| \leq \frac{1}{N} (2e)^{N-1} H^M 2^{N-M} \leq (2ed)^N$$

$d \geq 2 \leq d^{N-M}$

where **we ignore $\frac{1}{N}$ since we are going to use the log**

ERM on H_N \hat{h}

$$\ell_D(\hat{h}) \leq \min_{h \in H_N} \ell_D(h) + \sqrt{\frac{2}{m} \left(N \cdot (1 + \ln(2 \cdot d)) + \ln \frac{2}{\delta} \right)}$$

were $N \cdot (1 + \ln(2 \cdot d)) = \ln(H_N)$

In order to not overfit $m \gg N \cdot \ln d$

$N \cdot \ln d \ll 2^d$ for reasonable value of N

We grow the tree and a some point we stop.

$$\ell_D(h) \leq \hat{\ell}_S(h) + \varepsilon \quad \forall h \in H_N \quad \text{with probability at least } 1 - \delta$$

remove N in H_N and include h on ε

we remove the N index in H_N adding h on ε

$$\ell_D(h) \leq \hat{\ell}_S(h) + \varepsilon_h \quad \forall h \in H_N$$

$$W : H \rightarrow [0, 1] \quad \sum_{h \in H} w(h) \leq 1$$

How to use this to control over risk?

$$\mathbb{P} \left(\exists h \in H : |\hat{\ell}_S(h) - \ell_D(h)| > \varepsilon_h \right) \leq$$

where $\hat{\ell}_S$ is the prob my training set cases is true

$$\begin{aligned} &\leq \sum_{h \in H} \mathbb{P} \left(\left| \hat{\ell}_S(h) - \ell_D(h) \right| > \varepsilon_h \right) \leq \sum_{h \in H} 2e^{-2m\varepsilon_h^2} \leq \\ &\leq \delta \quad \rightarrow \text{since } w(h) \text{ sum to } 1 \left(\sum_{h \in H} \right) \end{aligned}$$

I want to choose $2e^{-2m\varepsilon_h^2} = \delta w(h)$

$$2e^{-2m\varepsilon_h^2} = \delta w(h) \quad \Leftrightarrow \quad \text{--- MANCA PARTEEEE ---}$$

therefore:

$$\ell_D(h) \leq \hat{\ell}_S(h) + \sqrt{\frac{1}{2m} \cdot \left(\ln \frac{1}{w(h)} + \ln \frac{2}{\delta} \right)} \quad \text{w. p. at least } 1 - \delta \quad \forall h \in H$$

Now, instead of using ERM we use

$$\hat{h} = \arg \min_{h \in H} \left(\hat{\ell}_S(h) + \sqrt{\frac{1}{2m} \cdot \left(\ln \frac{1}{w(h)} + \ln \frac{2}{\delta} \right)} \right)$$

where $\sqrt{\dots}$ term is the penalisation term

Since our class is very large we add this part in order to avoid overfitting. Instead of minimising training error alone i minimise training error + penalisation error.

In order to pick $w(h)$ we are going to use **coding theory**

The idea is I have my trees and i want to encode all tree predictors in H using strings of bits.

$\sigma : H \rightarrow \{0, 1\}^*$ **coding function for trees**
 $\forall h, h' \in H \quad \sigma(h)$ not a prefix of $\sigma(h')$
 $h \neq h' \quad$ where $\sigma(h)$ and $\sigma(h')$ are **string of bits**

σ is called **instantaneous coding function**

Instantaneous coding function has a property called **kraft inequality**

$$\sum_{h \in H} 2^{-|\sigma(h)|} \leq 1 \quad w(h) = 2^{-|\sigma(h)|}$$

I can design $\sigma : H \rightarrow \{0, 1\}^*$ instantaneous $|\sigma(h)|$

$$\ln |H_N| = O(N \cdot \ln d)$$

number of bits i need = number of node in h

Even if i insist in instantaneous i do not lose ... - MANCA PARTE -

$$|\sigma(h)| = O(N \cdot \ln d)$$

Using this σ and $w(h) = 2^{-|\sigma(h)|}$

$$\ell_D(h) \leq \hat{\ell}_S(h) + \sqrt{\frac{1}{2m} \cdot \left(c \cdot N \cdot \ln d + \ln \frac{2}{\delta} \right)} \quad w. p. \text{ at least } 1 - \delta$$

where c is a constant

$$\hat{h} = \arg \min_{h \in H} \left(\hat{\ell}_S(h) + \sqrt{\frac{1}{2m} \cdot \left(c \cdot N \cdot \ln d + \ln \frac{2}{\delta} \right)} \right)$$

where $m \gg N \cdot h \cdot \ln d$

If training set size is very small then you should not run this algorithm.

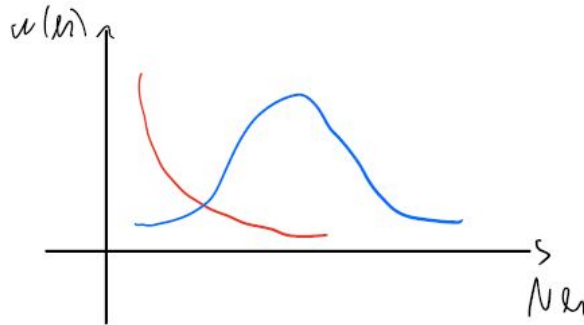


Figure 1.3: Algorithm for tree predictors

This blue curve is an alternative example. We can use Information criterion.

As I increase the number of nodes, N_h decrease so fast. You should take a smaller tree because it gives you a better bound. It's a principle known as Occam Razor (if I have two tree with the same error, if one is smaller than the other than i should pick this one).

Having N^*