# Statistical Learning Theory Exercise Handbook

Compiled by Matei Bejan
Faculty of Mathematics and Computer Science
University of Bucharest

February 2021

# Foreword

I wish for this work to be a statement.

A statement against all the oppression, the mistreatment and the dehumanisation we've been subjected to for too long.

A statement against the abject labour we are forced into, that of desperately searching for a match and a light into a labyrinth of ever-growing darkness, being ever mislead by the very people we are expected to see as our tutors.

It is time all Computer Science students, regardless of gender, colour, religion or creed, to be united into one strong voice and, like thunder in a roaring tempest, to strike directly at the heart of those keeping us in the dark.

For too long the authors of the great works of theoretical machine learning have forced us into such a grueling labour, to grind through their 1000-page tomes in order to find the one (or even two, if God wills it!) exercises with solutions.

For too long the prestigious, prodigious professors have prophesied fruitless exercises that oft as not help build up little to no intuition and experience in solving the most frequent problems in this field.

For too long have we desperately begged Google for help, frantically searching for answers that the very prophets of this field keep away from our minds.

For too long have we been misguided, have we been deliberately stirred away from the knowledge, the information, the intuition we need to solve those types of exercises mentioned above.

The tyranny of darkness ends today. The age of intentional ignorance comes to a close. And is it we that shall send it to its binary grave!

Rise, fellow CompSci students, they fear us!

They fear our intellect, so they stash away their prised solutions and give us convoluted proofs that are as helpful as jar of sand to a Saharan traveller!

They fear our ingenuity, so the few exercises we're being fed present all but little variation in topics!

They fear our adaptability, so they keep exaggerating the difficulty of their seminar work or book examples in order to confuse us!

Today, my friends, fear we shall give them! I speak to you all, oh students!, let your eyes feast on the exercises and solutions provided in this book and learn. Learn and show them once and for all that they cannot fool us into studying useless notions for our exams!

To those that have already pushed the boulders of Sisif and withstood the plight that are ML theory seminars, assignments and exams, I call thee to contribute to this work, to add more problems and solutions, more variation, more nuances of difficulty and even more sections!

An to you all, scornful professors and authors, those that deliberately keep us locked in the basement of our ignorance, you shall see us rise, learn and best you lot!

The Compiler

## Contributors

I would like to thank everyone who contributed to this work. If you've contributed via GitHub, please add your name (or at least your GitHub) to this list.

Marius Miculță-Câmpeanu, University of Bucharest, Romania.

Ștefan Mereuță, University of Bucharest, Romania.

Andrei Manolache, University of Bucharest, Romania.

Rob van Gastel, Eindhoven University of Technology, Netherlands.

# Purpose

The purpose of this hereby booklet is to centralize as many exercises and solutions to frequently asked exam and homework problems in the theoretical machine learning world. It is well known that computer science students usually lack formal, mathematical rigor, which they replace with what they call "common sense" or intuition. In order to help build that intuition, we've decided it would be best to create a central resource for problems, so that we can increase the sample size of the training data and thus help them achieve the best empirical loss.

Please do note that only a small portion of the solutions presented here are provided by myself. Most are provided either by contributors or by selecting solved problems over the internet or from well-known theory books, such as those of the Shais's or Mohri's. My work here was simply compiling them into one booklet, so that others waste not the time I have wasted looking over the internet for intuitive solutions.

At the time the first version of this book was written (February 2021), most of the exercises come through either the assignments presented by professor Bogdan Alexe, Ph.D. (Faculty of Mathematics and Computer Science, University of Bucharest) to his students or Shai Ben-David's and Shai Shalev-Shwarz's *Understanding Machine Learning* and Alon Gonen's solutions for their proposed exercises (note that I only took a few problems from there, which I considered to have the most intuitive solutions).

I did not consider it important to attribute the exercises and solutions to their respective providers, as the purpose of the booklet is not to help the contributors get acknowledged, but to help other computer science students study theoretical machine learning.

In the case the reader may have the resources and noble generosity to contribute, he may do so at the following GitHub address: https://github.com/mateibejan14/stl-exercise-handbook. Please do remember to update both the LaTeX code and PDF documents, as well as providing additional recourses such as images in case those are needed.

The Compiler (again)

# Contents

# 1 PAC-learnability

**Exercise 1.** An axis aligned rectangle classifier in the plane is a classifier that assigns the value 1 to a point if and only if it is inside a certain rectangle. Formally, given real numbers $a_1 \leq b_1$, $a_2 \leq b_2$, define the classifier $h_{(a_1,b_1,a_2,b_2)}$ by:

$$h_{(a_1,b_1,a_2,b_2)}(x_1, x_2) = \begin{cases} 1, & a1 \leq x_1 \leq b_1 \text{ and } a_2 \leq x_2 \leq b_2 \\ 0, & \text{otherwise} \end{cases}$$

The class of axis-aligned rectangles in the plane is defined as:'

$$\mathcal{H}^2_{rec} = \{h_{(a_1,b_1,a_2,b_2)} \mid a_1 \leq b_1, a_2 \leq b_2\}.$$

a) Prove that the class of axis-aligned rectangles is PAC-learnable and show that the learning algorithm provided is polynomial is $d$, $1/\epsilon$ and in $\log(1/\delta)$.

b) Prove that the algorithm you provided at *a)* has a sample complexity of $m_H(\epsilon, 1/\delta)$.

**Solution.**
a)

$$\mathcal{H}^2_{rec} = \left\{ \begin{array}{l} h_{(a_1,b_1,a_2,b_2)} \colon \mathbb{R}^2 \to \{0,1\}, a_1 \leq b_1 \text{ and } a_2 \leq b_2, \\ h_{(a_1,b_1,a_2,b_2)}(x_1, x_2) = \begin{cases} 1, & a1 \leq x_1 \leq b_1 \text{ and } a_2 \leq x_2 \leq b_2 \\ 0, & \text{otherwise} \end{cases} \end{array} \right\}$$

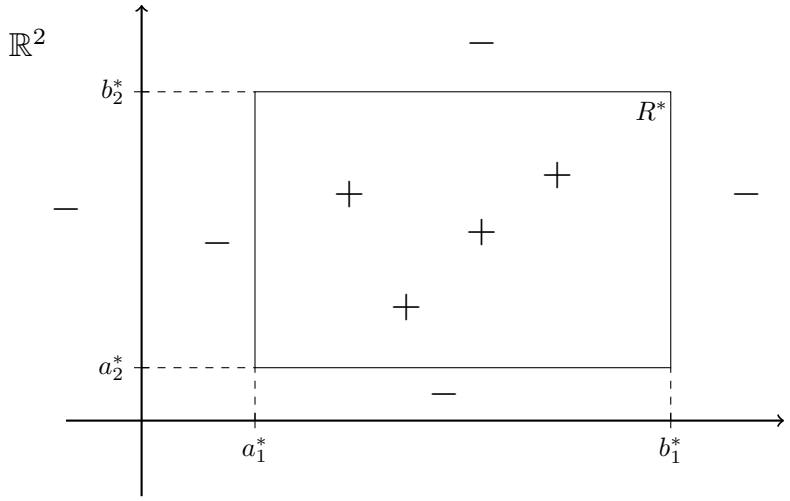
$\mathcal{H}^2_{rec}$ is an infinite size hypothesis class, it is called the class of all axis aligned rectangles in the plane. We want to prove that $\mathcal{H}^2_{rec}$ is PAC-learnable.

*Proof.* From the definition of PAC-learnability, we know that $\mathcal{H} = \mathcal{H}^2_{rec}$ is PAC-learnable if there exists a function $m_{\mathcal{H}} \colon (0,1)^2 \to \mathbb{N}$ and there exists a learning algorithm $A$ with the following property: for every $\epsilon, \delta > 0$, for every labelling function $f \in \mathcal{H}^2_{rec}$ (realizability case), for every distribution $\mathcal{D}$ on $\mathbb{R}^2$ when we run the learning algorithm $A$ on a training set $S$ consisting of $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ examples sampled i.i.d. from $\mathcal{D}$ and labeled by $f$, the algorithm $A$ returns a hypothesis $h_S \in \mathcal{H}$ such that, with probability at least $1 - \delta$ (over the choice of examples), the real risk of $h_S$ is smaller than $\epsilon$:

$$\begin{aligned} &\underset{S \sim \mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon) \geq 1 - \delta \text{ or otherwise said} \\ &\underset{S \sim \mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) > \epsilon) < \delta \end{aligned}$$

First, we need to find the algorithm $A$.

We are under the realizability assumption, so there exists a labelling function $f \in \mathcal{H}, f = h^*_{(a_1^*,b_1^*,a_2^*,b_2^*)}$ that labels the training data.




Consider the training set $S = \left\{ (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \;\middle|\; \begin{array}{l} y_i = h^*_{(a_1^*, b_1^*, a_2^*, b_2^*)}(x_i), \\ x_i \in \mathbb{R}^2, x_i = (x_{i1}, x_{i2}) \end{array} \right\}$

$h^*$ labels each point drawn from the rectangle $R^* = [a_1^*, b_1^*] \times [a_2^*, b_2^*]$ with label 1, and all other points with label 0. So we have $h^*_{(a_1^*, b_1^*, a_2^*, b_2^*)} = \mathbb{1}_{R^*}$

Consider the following algorithm $A$, that takes as input the training set $S$ and outputs $h_S$.

$h_S = h_{(a_{1S}, b_{1S}, a_{2S}, b_{2S})}$, where

$$a_{1S} = \min_{\substack{i=\overline{1,m} \\ y_i=1}} x_{i1} \qquad a_{2S} = \min_{\substack{i=\overline{1,m} \\ y_i=1}} x_{i2}$$

$$b_{1S} = \max_{\substack{i=\overline{1,m} \\ y_i=1}} x_{i1} \qquad b_{2S} = \max_{\substack{i=\overline{1,m} \\ y_i=1}} x_{i2}$$

If all $y_i = 0$, then all points $x_i$ have label 0, so there is no positive example. In this case, choose $z = (z_1, z_2)$ a point that is not in the training set $S$ and take $a_{1S} = b_{1S} = z_1$, $a_{2S} = b_{2S} = z_2$.

As in the ?indication, $h_S = h_{(a_{1S}, b_{1S}, a_{2S}, b_{2S})} = \mathbb{1}_{[a_{1S}, b_{1S}] \times [a_{2S}, b_{2S}]}$ is the indicator function of the tightest rectangle $R_S = [a_{1S}, b_{1S}] \times [a_{2S}, b_{2S}]$ enclosing all positive examples.

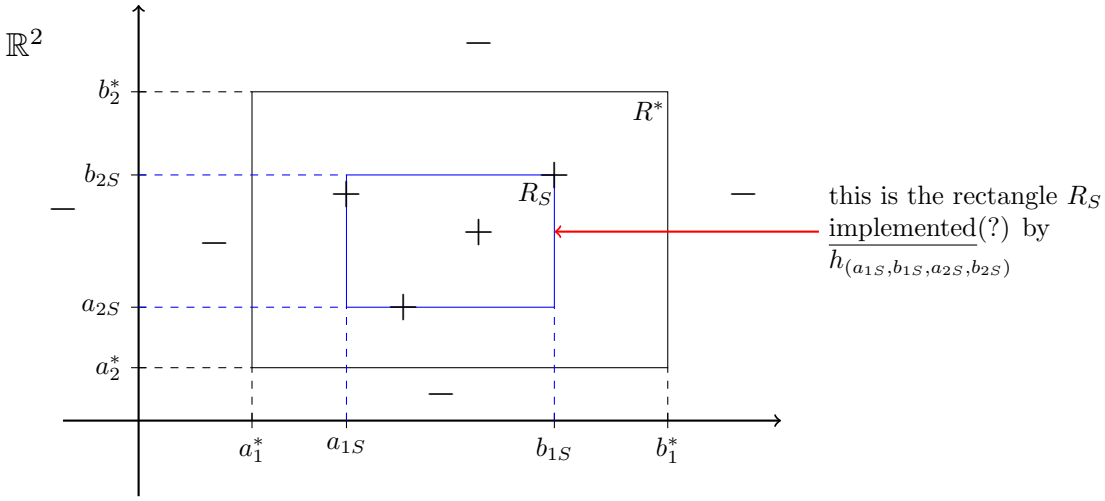
By construction, $A$ is an ERM, meaning that $L_{h^*, \mathcal{D}}(h_S) = 0$, $h_S$ doesn't make any errors on the training set $S$.

Now we want to find the sample complexity $m_{\mathcal{H}}(\epsilon, \delta)$ such that

$$\underset{S \sim \mathcal{D}^m}{P}(L_{h^*, \mathcal{D}}(h_S) \leq \epsilon) \geq 1 - \delta \text{ where } S \text{ contains } m \geq m_{\mathcal{H}}(\epsilon, \delta) \text{ examples.}$$

We make the observation that $h_S$ can make errors in region $R^* \setminus R_S$, labelling points that should get label 1 with label 0. All points $\in R^*$[1] will be labeled correctly, all points outside $R^*$ will be labeled correctly.

[1] $R_S$?




Let's fix $\epsilon > 0, \delta > 0$ and consider a distribution $\mathcal{D}$ over $\mathbb{R}^2$.

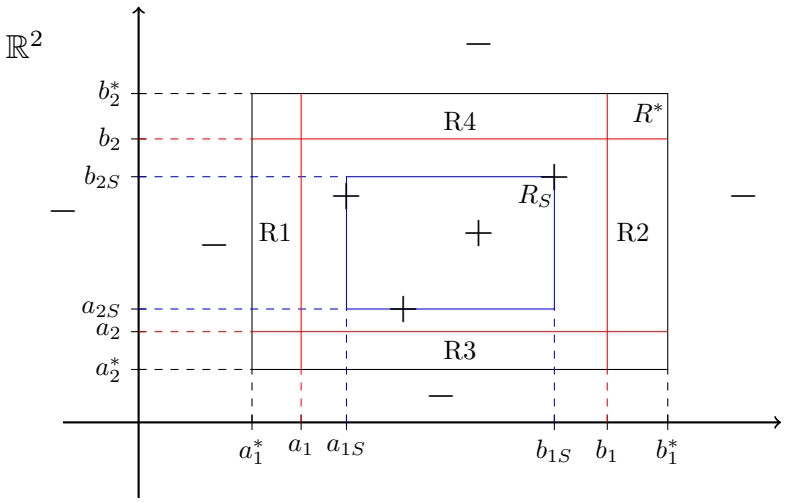
Case 1)

If $\mathcal{D}(R^*) = \underset{x\sim\mathcal{D}}{P}(x \in R^*) \leq \epsilon$ then in this case

$L_{h^*,\mathcal{D}}(h_S) = \underset{x\sim\mathcal{D}}{P}(h_S(x) \neq h^*(x)) = \underset{x\sim\mathcal{D}}{P}(x \in R^* \setminus R_S) \leq \underset{x\sim\mathcal{D}}{P}(x \in R^*) \leq \epsilon$ so we have that

$\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) = 1$ (this happens all the time)

Case 2) $\mathcal{D}(R^*) = \underset{x\sim\mathcal{D}}{P}(x \in R^*) > \epsilon$



We construct as in the indication the 4 rectangles $R_1, R_2, R_3, R_4$:

$$\begin{array}{ll} R_1 = [a_1^*, a_1] \times [a_2^*, b_2^*] & R_2 = [b_1, b_1^*] \times [a_2^*, b_2^*] \\ R_3 = [a_1^*, b_1^*] \times [a_2^*, a_2] & R_4 = [a_1^*, b_1^*] \times [b_2, b_2^*] \end{array} \qquad \text{with } \mathcal{D}(R_i) = \underset{x \sim \mathcal{D}}{P}(x \in R_i) = \frac{\epsilon}{4}$$

If $R^2 = [a_{1S}, b_{1S}] \times [a_{2S}, b_{2S}]$ (the rectangle returned by $A$, implemented by $h_S$) intersects each $R_i, i = \overline{1,4}$:

$$L_{h^*,\mathcal{D}}(h_S) = \underset{x \sim \mathcal{D}}{P}(h^*(x) \neq h_S(x)) = \underset{x \sim \mathcal{D}}{P}(x \in R^* \setminus R_S) \leq \underset{x \sim \mathcal{D}}{P}(x \in R_1 \cup R_2 \cup R_3 \cup R_4) \leq$$

$$\leq \sum_{i=1}^{4} \underset{x \sim \mathcal{D}}{P}(x \in R_i) = \sum_{i=1}^{4} \mathcal{D}(R_i) = 4 \cdot \frac{\epsilon}{4} = \epsilon$$

So, in this case, $\underset{S \sim \mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) = 1$ (this happens always).

In order to have $L_{h^*,\mathcal{D}}(h_S) > \epsilon$, we need that $R_S$ will not intersect at least one rectangle $R_i$.

We denote with $F_i = \{S \sim \mathcal{D}^m \mid R_S \cap R_i = \varnothing\}$. This leads to the following:

$$\underset{S \sim \mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon) = \underset{S \sim \mathcal{D}^m}{P}(\overset{\text{at least one } F_i \text{ will happen}}{\nearrow}(F1 \cup F2 \cup F3 \cup F4) \leq \sum_{i=1}^{4} \underset{S \sim \mathcal{D}^m}{P}(F_i)$$

Now, $\underset{S \sim \mathcal{D}^m}{P}(F_i) =$ what is the probability that $R_S$ will not intersect $R_i$

$$= \text{the probability that no point from } R_i \text{ is sampled in } S$$

$$= \left(1 - \frac{\epsilon}{4}\right)^m$$

So

$$\underset{S \sim \mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon) \leq \sum_{i=1}^{4} \underset{S \sim \mathcal{D}^m}{P}(F_i) = 4 \cdot \left(1 - \frac{\epsilon}{4}\right)^m$$

Now, we know from lecture 2 that $1 - x \leq e^{-x}$, so $1 - \frac{\epsilon}{4} \leq e^{-\frac{\epsilon}{4}}$, which means that

$$\underset{S \sim \mathcal{D}^m}{P}(\underset{\substack{\uparrow \\ \text{this is the probability to have} \\ h_S \text{ will make(??) error} > \epsilon}}{L_{h^*,\mathcal{D}}(h_S) > \epsilon}) \leq 4 \cdot \left(1 - \frac{\epsilon}{4}\right)^m \leq 4 \cdot e^{-\frac{\epsilon}{4}m}$$

We want to make this probability very small, smaller than $\delta$:

$$4 \cdot e^{-\frac{\epsilon}{4}m} < \delta$$

$$e^{-\frac{\epsilon}{4}m} < \frac{\delta}{4} \quad \Big| \cdot \log_e$$

$$-\frac{\epsilon}{4} \cdot m < \log \frac{\delta}{4} \quad \Big| \cdot \left(-\frac{4}{\epsilon}\right)$$

$$m > -\frac{4}{\epsilon} \log \frac{\delta}{4} = \frac{4}{\epsilon} \log \frac{4}{\delta}$$

[2] $R_S$?

So, if we take $m \geq m_{\mathcal{H}}(\epsilon, \delta) = \frac{4}{\epsilon} \cdot \log \frac{4}{\delta}$, we obtain the desired results.

Repeat the previous question for the class of aligned rectangles in $\mathbb{R}^d$.
In $\mathbb{R}^d$, we have

$$\mathcal{H}^d_{rec} = \left\{ \begin{array}{l} h_{(a_1,b_1,a_2,b_2,\ldots,a_d,b_d)} \colon \mathbb{R}^d \to \{0,1\} \mid a_i \leq b_i, i = \overline{1,d} \\ h_{(a_1,b_1,a_2,b_2,\ldots,a_d,b_d)} = \mathbb{1}_{[a_1,b_1]\times[a_2,b_2]\times\cdots\times[a_d,b_d]} \end{array} \right\}$$

All the arguments used previously will work, the general result will be that $m_{\mathcal{H}}(\epsilon, \delta) = \frac{2d}{\epsilon} \cdot \log \frac{2d}{\delta}$.
For $d = 2$, we obtain the previous result.

The runtime of algorithm $A$ is given by taking ? minimum[3] over each dimension, so this means $\mathcal{O}(m*d)$: $m$ = number of (positive) examples = $\mathcal{O}(\frac{2d}{\epsilon} \cdot \log \frac{2d}{\delta})$
$d$ = number of dimensions.

So we have that the complexity of algorithm $A$ is $\mathcal{O}(\frac{2d^2}{\epsilon} \cdot \log \frac{2d}{\delta})$, which is polynomial in $d$, $\frac{1}{\epsilon}$, $\frac{1}{\delta}$.

□

b)

*Proof.* $\mathcal{H}$ is PAC-learnable with sample complexity $m_{\mathcal{H}}(\cdot, \cdot)$ means that there exists a learning algorithm $A$ with the following property: for every $\epsilon, \delta > 0$, when we run the algorithm $A$ on a sample set $S$ of $m$ examples, $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ (samples are labeled by $f \in \mathcal{H}$ and i.i.d. from a distribution $\mathcal{D}$), we have that $h_S = A(S)$ with the real risk $\underset{S\sim\mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon) > 1-\delta$.

We apply this for $\epsilon_1$ and $\delta$:

$$\underset{S\sim\mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon_1) > 1 - \delta \text{ if } m \geq m_{\mathcal{H}}(\epsilon_1, \delta)$$

We know that $\epsilon_2 \geq \epsilon_1$, so we have that

$$\underset{S\sim\mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon_2) > 1 - \delta \text{ if } m \geq m_{\mathcal{H}}(\epsilon_1, \delta)$$

But $m_{\mathcal{H}}(\epsilon_2, \delta)$ is the smallest number of examples for which the above inequality holds. So, if it holds for $m \geq m_{\mathcal{H}}(\epsilon_1, \delta)$, we have that $m_{\mathcal{H}}(\epsilon_1, \delta) \geq m_{\mathcal{H}}(\epsilon_2, \delta)$. □

a) Given $\epsilon \in (0,1)$, $0 < \delta_1 \leq \delta_2 < 1$, we have that $m_{\mathcal{H}}(\epsilon, \delta_1) \geq m_{\mathcal{H}}(\epsilon, \delta_2)$.

*Proof.* Using the same arguments from **??**, we have that

$$\underset{S\sim\mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon) > 1 - \delta_1 \text{ if } m \geq m_{\mathcal{H}}(\epsilon, \delta_1)$$

$$\delta_1 \leq \delta_2 \Rightarrow 1 - \delta_1 \geq 1 - \delta_2 \Rightarrow \underset{S\sim\mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon) > 1 - \delta_2 \text{ if } m \geq m_{\mathcal{H}}(\epsilon, \delta_1)$$

[3] min,max?

But $m_{\mathcal{H}}(\epsilon, \delta_2)$ is the smallest number of examples for which the above inequality holds
(if $m \geq m_{\mathcal{H}}(\epsilon, \delta_2)$). So, if it holds for $m \geq m_{\mathcal{H}}(\epsilon, \delta_1)$, we have that $m_{\mathcal{H}}(\epsilon, \delta_1) \geq m_{\mathcal{H}}(\epsilon, \delta_2)$. □

**Exercise 2.** Consider H to be the class of all centered in origin sphere classifiers in the 3D space. A centered in origin sphere classifier in the 3D space is a classifier $h_r$ that assigns the value 1 to a point if and only if it is inside the sphere with radius $r > 0$ and center given by the origin O(0,0,0). Consider the realizability assumption.

Show that H can be $(\epsilon, \delta) - PAC$ learned by giving an algorithm A and determining the sample complexity $m_H(\epsilon, \delta)$ such that the definition of PAC-learnability is satisfied.

**Solution.**

The classifiers described in the problem are formally closed center-origin balls in $\mathbb{R}^3$. We can rewrite H as follows:

$$H = \{B[0, r] | r \in \mathbb{R}_+\} \subset \mathbb{R}^3$$

Let A be the following algorithm:

```
if ∃s ∈ S ⊂ ℝ² × {label | label ∈ {0,1}} with label 1:
    r = max_{label(s) = 1} ||s||
else:
    r = min_{label(s) = 0} ||s|| − 10^{−1000}
return B[0, r]
```

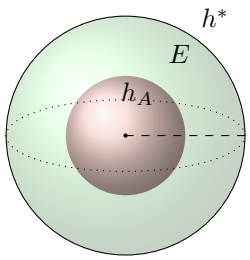
In other words, our algorithm A returns the minimal-volume sphere that contains all 1-labelled samples. In case no such samples exist, we return the sphere with the diameter smaller than the closest 0-labeled point to the origin.

We denote with $h_A, r_A$ the hypothesis returned by our algorithm and its associated radius, respectively. By assuming realizability, we denote with $h^*$ the hypothesis with zero error and with $r^*$, its radius.

Let $\epsilon, \delta \in (0, 1)$ and $r' \in \mathbb{R}, r' \leq r^*$, such that $D[x | r' \leq ||x|| \leq r^*] = \epsilon$. Next, let $E = B[0, r^*] \setminus B[0, r']$ represent the error area between our classifier and the zero-error classifier.

$\underset{S \sim D^m}{P}[L_D(h_S) \geq \epsilon] < P[\nexists x | x \in S \setminus E] < (1 - \epsilon^m) < e^{-\epsilon m}$

We want $e^{-\epsilon m} < \delta \Rightarrow m > \frac{\log \delta}{\epsilon}$.




**Exercise 3.** Let X be a discrete domain, and let $H_{Singleton} = h_z | z \in \chi \cup h^-$, where for each $z \in \chi$ , $h_z$ is the function defined by $h_z(x) = 1$ if x = z and $h_z(x) = 0$ if $x \neq z$. $h^-$ is simply the all-negative hypothesis, namely, $\forall x \in X, h^-(x) = 0$. The realizability assumption here implies that the true hypothesis f labels negatively all examples in the domain, perhaps except one.

a. Describe an algorithm that implements the ERM rule for learning $H_{Singleton}$ in the realizable setup.

b. Show that $H_{Singleton}$ is PAC learnable. Provide an upper bound on the sample complexity.

**Solution.**
$\mathcal{X}$ discrete domain, $\mathcal{H}_{\text{Singleton}} = \{h_z \colon z \in \mathcal{X}\} \cup \{h^-\}$

$$\forall z \in \mathcal{X} \quad h_z \colon \mathcal{X} \to \{0,1\}, \quad h_z(x) = \begin{cases} 1, & x = z \\ 0, & x \neq z \end{cases}$$
$$h^- \colon \mathcal{X} \to \{0,1\}, \quad h^-(x) = 0, \ \forall x \in \mathcal{X}$$

**a)** Describe an algorithm that implements the ERM rule for learning $\mathcal{H}_{\text{Singleton}}$ in the realizable setup.

*Proof.* Consider $S = \{(x_i, h^*(x_i)), x_i \text{ i.i.d. from a distribution } \mathcal{D} \text{ over } \mathcal{X}\}_{i=1}^m$.
The algorithm $A$ is the following:
Loop over training examples
If there is an $i \in \{1, \dots, m\}$ such that $y_i = 1$, then return hypothesis $h_S = A(S) = h_{x_i}$
Otherwise return $h^-$.
From construction, $A$ is ERM, meaning that $L_S(h_S) = 0$. □

**b)** Show that $\mathcal{H}_{\text{Singleton}}$ is PAC-learnable. Provide an upper bound on the sample complexity.

*Proof.* Let $\epsilon, \delta > 0$ and fix a distribution $\mathcal{D}$ over $\mathcal{X}$.
The only case in which the algorithm $A$ fails is the case where $h^* = h_z$ and the sample
$S = \{(x_i, y_i) \mid ?\ x_i \text{ sampled i.i.d. from } \mathcal{D}\}$ doesn't contain any positive examples, so all $y_i = 0\ \forall i = \overline{1,m}$.
In this case, $h_S = A(S) = h^-$, which is different from $h^*$. However, even if the algorithm $A$ fails if $\mathcal{D}(\{z\}) \leq \epsilon$, then everything is ok, as we have that:

$$\underset{S \sim \mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) = 1 \quad \checkmark\checkmark$$

So, we have to upper bound the sample complexity in the case where $\mathcal{D}(\{z\}) > \epsilon$ and there is no positive example in the set $S$ (actually, for this problem, there

is just one positive possible training point = $z$). We have that

$\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon)$ = probability that each point in $S$
is different than $z$ (which has probability mass $> \epsilon$) $\leq (1-\epsilon)^m \leq e^{-\epsilon m}$
So, if we set $e^{-\epsilon m} < \delta \Rightarrow -\epsilon m < \log\delta$

$$m > -\frac{1}{\epsilon}\log\delta \Rightarrow m > \frac{1}{\epsilon}\log\frac{1}{\delta}$$

If $m \geq \left\lceil \frac{1}{\epsilon}\log\frac{1}{\delta} \right\rceil$ we have that $\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon) < \delta$

So the upper bound of $m_{\mathcal{H}}(\epsilon,\delta)$ is $m_{\mathcal{H}}(\epsilon,\delta) \leq \left\lceil \frac{1}{\epsilon}\log\frac{1}{\delta} \right\rceil$

□

**Exercise 4.** An axis aligned square classifier in the plane is a classifier that assigns the value 1 to a point if and only if it is inside a certain square. Formally, given the real numbers $a_1, a_2, r > 0 \in \mathbb{R}$ we define the classifier $h_{(a_1,a_2,r)}$ by.

$$h_{a_1,a_2,r}(x_1,x_2) = \begin{cases} 1 & a_1 \leq x_1 \leq a_1 + r, a_2 \leq x_2 \leq a_2 + r \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

the class of all axis aligned squares in the plane is defined as

$$\mathcal{H} = \{h_{a_1,a_2,r} : \mathbb{R}^2 \to \{0,1\} | a_1, a_2, r \in \mathbb{R}, r > 0\}$$

Consider the realizability assumption.

a) Give a learning algorithm $\mathcal{A}$ that returns a hypothesis $h_S = \mathcal{A}(S)$ that has empirical risk 0 on $S$.

b) Find the sample complexity $m_{\mathcal{H}}(\varepsilon,\delta)$ in order to show that $\mathcal{H}$ is PAC-learnable.

**Solution.**

a) Consider the following ERM algorithm $\mathcal{A}$: Let $S_p$ be the set of all positive labeled points in the sample $S$. Project all the points in $S_p$ on the $Ox$ axis and let $S_{px}$ be the set of scalars obtained by taking the $x$ component of the projection, pick $a_1^* = min(\{S_{px}\})$ and $r_x = max(\{S_{px}\})$, do the same on the $Oy$ axis and build the $S_{py}$ set, pick $a_2^* = min(\{S_{py}\})$ and $r_y = max(\{S_{py}\})$. Let $r^* = max(\{||a_1^* - r_x||_2, ||a_2^* - r_y||_2\})$, pick the hypothesis $h_{a_1^*,a_2^*,r^*}$, which is the square that most "tightly" contains all the positive examples, so it is correctly labeling them.

If the obtained square has 0-labeled points from $S$ inside it, translate it on the $Ox$ and $Oy$ directions until there is no 0-labeled point inside - the realizability assumption guarantees that there exists a square $\tilde{R}$ corresponding to the real labeling function that perfectly labels all the points, so for our minimum enclosing square, we'll either eventually get a square that's included in $\tilde{R}$, so it is ERM, or a square that is just partially included in $\tilde{R}$ but it still is ERM (since it has no more negative points from the training set inside).

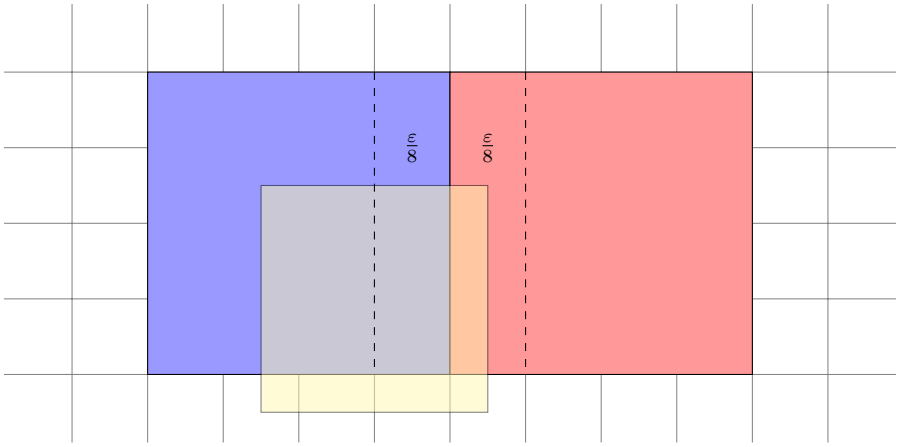



Figure 1: Blue: the optimal square $\tilde{R}$; Red: the square $SQ$ - same area as $\tilde{R}$; Transparent yellow: $\mathcal{A}(S)$, the errors from landing below $\tilde{R}$ are taken into account by the way $SQ$ is constructed.

b) The proof is very similar to the axis-aligned rectangles problem. Let $R(x, y, z, h)$ be the rectangle as defined in the axis aligned rectangles problem and $R(x, y, z)$ be a the rectangle $R(x, x + z, y, y + z)$.

Fix a distribution $\mathcal{D}$ over $\mathcal{X}$ and let $h_{\tilde{a_1},\tilde{a_2},\tilde{r}} = \tilde{h}$ be the hypothesis with 0 generalization error and $R(\tilde{a_1}, \tilde{a_2}, \tilde{r}) = \tilde{R}$ the associated square. Let $R(S)$ be the square returned by the ERM algorithm $\mathcal{A}$ for the training sequence $S$.

Let $\varepsilon, \delta \in (0, 1)$, and let $a_{12} > \tilde{a_1}$ such that $R_1 = R(\tilde{a_1}, a_{12}, \tilde{a_2}, \tilde{a_2} + \tilde{r})$ has probability mass $\frac{\varepsilon}{8}$ w.r.t. the distribution $\mathcal{D}$. In a similar fashion, get 3 more rectangles $R_2 = R(a_{11}, \tilde{a_1} + \tilde{r}, \tilde{a_2}, \tilde{a_2} + \tilde{r}), R_3 = R(\tilde{a_1}, \tilde{a_1} + \tilde{r}, a_{21}, \tilde{a_2} + \tilde{r}), R_4 = R(\tilde{a_1}, \tilde{a_1} + \tilde{r}, \tilde{a_2}, a_{22})$ such that every one of them has mass $\frac{\varepsilon}{8}$.

Until now, the reasoning was identical to that of the axis-aligned rectangles (except for the probability mass of $\frac{\epsilon}{8}$). Now construct one more square as below:

The square $SQ$ in the right has the same mass as the optimal square $\tilde{R}$. Construct 4 more rectangles $R_5 \dots R_8$ with probability mass $\frac{\varepsilon}{8}$ w.r.t. $\mathcal{D}$ on $SQ$ the same way we did on $\tilde{R}$.

We don't care about the other 7 squares (on the diagonals, in the left or the ones on top and bottom), we take into account possible errors from them on $SQ$ since we're assuming realizability and the total area of the square returned from $\mathcal{A}$ can be at most as big as that of $\tilde{R}$, and the biggest error appears if $\mathcal{A}$ returns one of the squares like $SQ$ (if it sees just 2 corner points on one border of $\tilde{R}$).

We have that $\mathcal{L}_{(\mathcal{D},\tilde{h})}(R(S)) = \mathcal{D}((\tilde{R} \backslash R(S)) \cup (R(S) \cap SQ))$.

Now, for $\forall i \in \overline{1, 8}$ define $F_i = \{S|_x | S|_x \cap R_i = \varnothing\}$; for a sequence of $m$ points we have that

$$\mathcal{D}^m(\{S|\mathcal{L}_{(\mathcal{D},\tilde{h})}(\mathcal{A}(S)) > \varepsilon\}) \leq \mathcal{D}^m(\bigcup_{i\in\overline{1,8}} F_i) \tag{2}$$

By applying Boole's inequality we get that

$$\mathcal{D}^m(\bigcup_{i\in\overline{1,8}} F_i) \leq \sum_{i\in\overline{1,8}} \mathcal{D}^m(F_i) \tag{3}$$

Now take an arbitrary, but fixed $i \in \overline{1,8}$ - if we have a sample in $F_i$ we know that there's some sample that is not in $R_i$, so $\mathcal{D}^m(F_i) = (1 - \frac{\varepsilon}{8})^m \leq e^{-m\frac{\varepsilon}{8}}$. Combining (2) and (3) we get that

$$\mathcal{D}(\{S|\mathcal{L}_{(\mathcal{D},\tilde{h})}(\mathcal{A}(S)) > \varepsilon\}) \leq 8e^{-m\frac{\varepsilon}{8}} \tag{4}$$

Now we can see that as our sample size grows, the probability to have errors gets smaller, pick $\delta \geq 8e^{-m\frac{\varepsilon}{8}}$, we compute that $-m\frac{\varepsilon}{8} \leq log(\frac{\delta}{8}) \Rightarrow -m \leq \frac{8}{\varepsilon}log(\frac{\delta}{8}) \Rightarrow m \geq \frac{8}{\varepsilon}log(\frac{8}{\delta})$.

**Exercise 5 .** An axis aligned square classifier in the plane is a classifier that assigns the value 1 to a point if and only if it is inside a certain square. Formally, given the real numbers $a_1, a_2, r > 0 \in \mathbb{R}$ we define the classifier $h_{(a_1,a_2,r)}$ by

$$h_{(a_1,a_2,r)}(x_1, x_2) = \begin{cases} 1 & \text{if } a_1 \leq x_1 \leq a_1 + r, a_2 \leq x_2 \leq a_2 + r \\ 0 & \text{otherwise} \end{cases}$$

The class of all axis-aligned squares in the plane is defined as

$$\mathcal{H} = \{h_{(a_1,a_2,r)} \colon \mathbb{R}^2 \to \{0,1\} \mid a_1, a_2, r \in \mathbb{R},\ r > 0\}$$

Consider the realizability assumption.

**a.** give a learning algorithm $A$ that returns a hypothesis $h_S$ from $\mathcal{H}$, $h_S = A(S)$ consistent with the training set $S$ ($h_S$ has empirical risk 0 on $S$);

**b.** find the sample complexity $m_{\mathcal{H}}(\epsilon, \delta)$ in order to show that $\mathcal{H}$ is PAC-learnable;

**c.** compute $VC\dim\mathcal{H}$.

**Solution**.

*Proof.* Consider the realizability assumption: there $\exists\ h_{(a_1^*,a_2^*,r^*)}$ labeling the training data.

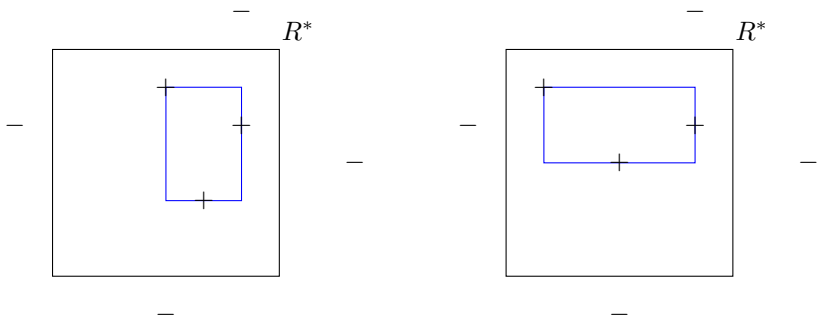
a) $S = \{(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_m, y_m)\}$, $y_i = h_{(a_1^*,a_2^*,r^*)}(x_i)$.

We need to find $a_{1S}, a_{2S}, r_S$ such that $h_S = A(S) = h_{a_{1S},a_{2S},r_S}$ is ERM.

Sketch of proof

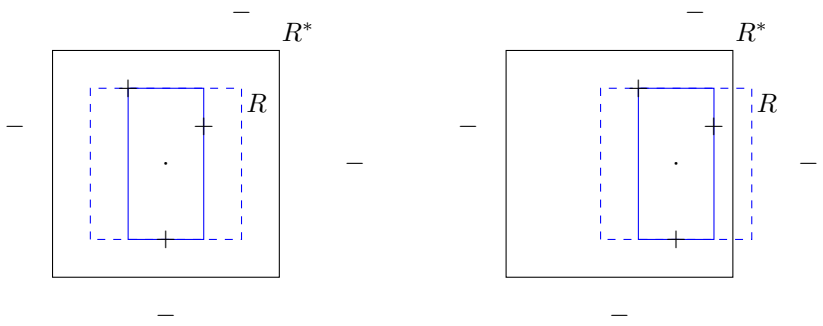
Particular cases: two positive or just 1 positive point in $S$.

General case: more than 2 positive points in $S \rightarrow$ find the smallest rectangle containing all positive points. We need to extend this rectangle such that we obtain a square.




Parameterize the rectangle by center $(x_0, y_0)$, width $w$, height $h$.

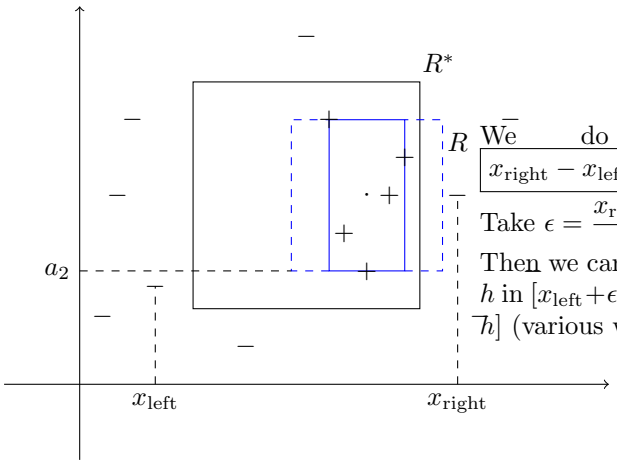
- if $w = h$ OK (we have a square)
- without loss of generality, we can consider $h \geq w$.




(could extend the initial rectangle to a square outside $R^*$)

Use the following steps:

1. check if the square with center $(x_0, y_0)$ and side $= h$ contains a negative point. If not, return this square.
2. If the previous constructed square contains negative points from $S$, then:
   - find $x_{\text{left}}$ = the largest value on the 1st coordinate of a negative point in the left part of our rectangle; if $x_{\text{left}}$ does not exist, OK $\checkmark \rightarrow$ we can build the square to the left
   - find $x_{\text{right}}$ = the smallest value on the 1st coordinate of a negative point in the right part of our initial rectangle; if there is no such negative point, OK $\Rightarrow$ we can build our desired square in the right



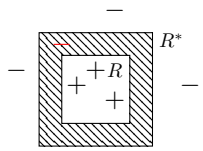

We do know that $\boxed{x_{\text{right}} - x_{\text{left}} > h}$

Take $\epsilon = \dfrac{x_{\text{right}} - x_{\text{left}} - h}{2} > 0$.

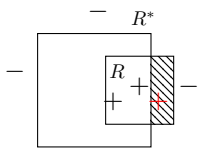
Then we can fit a square of side $h$ in $[x_{\text{left}} + \epsilon, x_{\text{right}} - \epsilon] \times [a_2, a_2 + h]$ (various ways to do this)

b) Two types of errors:

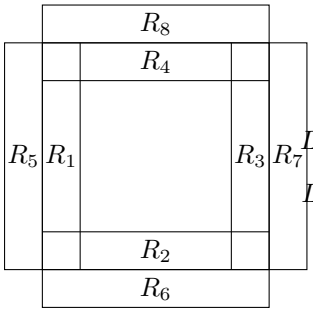
- false negatives in $R^* \setminus R$




- false positives in $R \setminus R^*$




Let $\epsilon > 0$, $\delta > 0$ and $\mathcal{D}$ a probability distribution on $\mathbb{R}^2$.
Consider regions of probability mass $= \frac{\epsilon}{8}$.




$$L_{\text{FN},i}(h_S) = \underset{x \sim \mathcal{D}}{P}[x \in R_i] \qquad i = \overline{1,4}$$
$$L_{\text{FP},i}(h_S) = \underset{x \sim \mathcal{D}}{P}[x \in R_{i+4}] \quad i = \overline{1,4}$$

So, if we want that $L_{\mathcal{D},h^*}(h_S) > \epsilon$, then one of the eight losses $\geq \frac{1}{8}$.

$$\text{Define } F_i = \left\{ S \sim \mathcal{D}^m \mid L_{\text{FN},i}(h_S) \geq \frac{1}{8} \right\} \quad i = \overline{1,4}$$

$$F_{i+4} = \left\{ S \sim \mathcal{D}^m \mid L_{\text{FP},i}(h_S) \geq \frac{1}{8} \right\} \quad i = \overline{1,4}$$

$$\text{So } \underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) \leq \underset{S\sim\mathcal{D}^m}{P}(F_1 \cup \cdots \cup F_8) \leq \sum_{i=1}^{8} P(F_i) = 8 \cdot \left(1 - \frac{\epsilon}{8}\right)^m \leq 8 \cdot e^{-m\cdot\frac{\epsilon}{8}} < \delta$$

$$\Rightarrow \boxed{m > \frac{8}{\epsilon} \log \frac{8}{\delta}}$$

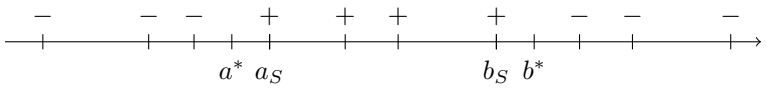
□

**Exercise 6.** Give a PAC-learning algorithm for the concept class $\mathcal{C}$ formed by closed intervals [a,b] with $a, b \in \mathbb{R}$.

**Solution.**

$$\mathcal{C} = \mathcal{H}_{\text{intervals}} = \left\{ \begin{array}{l} h_{a,b} \colon \mathbb{R} \to \{0,1\}, a \leq b \\ h_{a,b}(x) = \mathbb{1}_{[a,b]}(x) = \left\{ \begin{array}{l} 1, x \in [a,b] \\ 0, \text{otherwise} \end{array} \right. \end{array} \right\}$$

Consider a training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$. We are in the realizability case, $\exists h^* = h_{a^*,b^*} \in \mathcal{H}_{\text{intervals}}$ that labels the examples, $y_i = h^*(x_i)$.




We want to show that $\mathcal{H}_{\text{intervals}}$ is PAC-learnable.

Consider $A$ the learning algorithm that gets the training set $S$ and outputs $h_S = A(S) =$ the tightest interval containing all the positive examples.

$$h_S = h_{a_S,b_S} \qquad \text{where} \qquad a_S = \min_{(x_i,1)\in S} x_i \qquad b_S = \max_{(x_i,1)\in S} x_i \qquad R_S = [a_S, b_S]$$

If there is no $(x_i, 1) \notin S$ ($S$ doesn't contain positive examples), take $a_S = b_S = z$ a random point such that $(z, 0) \notin S$.

From construction, we see that $L_S(h_S) = 0$.

Let $\epsilon > 0, \delta > 0$ and $\mathcal{D}$ a distribution over $R$. How many training examples $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ do we need such that

$$\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) < \delta$$


Case 1: if $\mathcal{D}([a^*, b^*]) \leq \epsilon$ then $\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) = 0$ ✓

Case 2: if $\mathcal{D}([a^*, b^*]) > \epsilon$

Build $R_1$ and $R_2$, $R_1 = [a_1^*, a_1]$, $R_2 = [b_1, b_1^*]$ such that $\mathcal{D}(R_1) = \mathcal{D}(R_2) = \frac{\epsilon}{2}$.

If $R_S \cap R_1 \neq \varnothing$ and $R_S \cap R_2 \neq \varnothing$ then $\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) = 0$ ✓

Else $\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) \leq 2 \cdot \left(1 - \frac{\epsilon}{2}\right)^m \leq 2 \cdot e^{-\frac{\epsilon}{2}m} < \delta \Rightarrow \boxed{m > \frac{2}{\epsilon}\log\frac{2}{\delta}}$

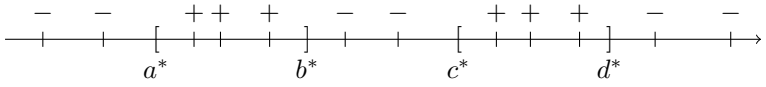
**Exercise 7.** Give a PAC-learning algorithm for the concept class $\mathcal{C}_2$ formed by unions of two closed intervals, that is $[a, b] \cup [c, d]$, with $a, b, c, d \in \mathbb{R}$. Extend your result to derive a PAC-learning algorithm for the concept class $\mathcal{C}_p$ formed by unions of $p \geq 1$ closed intervals, thus $[a_1, b_1] \cup \cdots \cup [a_p, b_p]$, with $a_k, b_k \in \mathbb{R}, \forall k \in [p]$. What are the time and sample complexities of your algorithm as a function of p?

**Solution.**

PAC-learning algorithm for the class $\mathcal{C}_2$ formed by unions of two closed intervals:

$$\mathcal{C}_2 = \left\{ \begin{array}{l} h_{(a,b,c,d)} : \mathbb{R} \to [0,1],\ h_{(a,b,c,d)} = \mathbb{1}_{[a,b]\cup[c,d]} \\ a \leq b \leq c \leq d,\ h_{(a,b,c,d)}(x) = \left\{ \begin{array}{l} 1, x \in [a,b] \cup [c,d] \\ 0, \text{otherwise} \end{array} \right. \end{array} \right\}$$

Consider $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, where $y_i = h^*(x_i), h^* = h_{(a^*,b^*,c^*,d^*)}$ [4].




Consider the following learning algorithm $A$ that takes input $S$:

i) Sort $S$ in ascending order of $x_i$

ii) Go over the sorted training examples and take the intervals where consecutive training examples labeled as positive start and end the intervals. You can obtain one or two intervals. In the case that there are no positive examples take $a_S = b_S = c_S = d_S = z$, where $z$ a random points such that $(z, 0)$ doesn't appear in the training set $S$.

iii) If you obtained just one interval, you can have $a_S = \min_{\substack{(x_i,y_i)\\ y_i=1}} x_i$, $b_S = \max_{\substack{(x_i,y_i)\\ y_i=1}} x_i$, $c_S = d_S = b_S$

[4] realizability assumption

If you obtained two intervals, then $a_S = \min\limits_{\substack{(x_i,y_i)\\ y_i=1}} x_i$, $d_S = \max\limits_{\substack{(x_i,y_i)\\ y_i=1}} x_i$, $a_S \le b_S < c_S \le d_S$

Return $h_S = h_{(a_S,b_S,c_S,d_S)} = \mathbb{1}_{[a_S,b_S]\cup[c_S,d_S]}$.

We need to find $m \ge m_{\mathcal{C}_2}(\epsilon,\delta)$ such that for $\epsilon > 0$, $\delta > 0$ and for every $\mathcal{D}$ distribution over $\mathbb{R}$ we have

$$\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) < \delta$$

Let $\epsilon > 0$, $\delta > 0$ and let $\mathcal{D}$ be a distribution over $\mathbb{R}$.

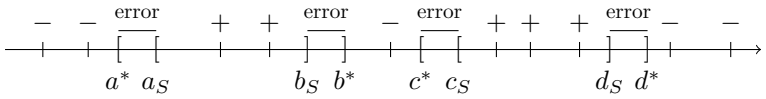
The region where $h_S$ can make errors is always $\subseteq [a^*, d^*]$.

Case 1: If $\mathcal{D}([a^*, d^*]) \le \epsilon$, then $\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S) > \epsilon) = 0$.

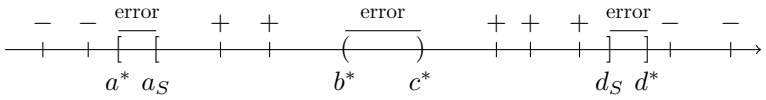
Case 2: If $\mathcal{D}([a^*, d^*]) > \epsilon$

The types of error that $h_S$ can make are:

- false negatives in $[a^*, b^*]$ and $[c^*, d^*]$




- false positive in $(b^*, c^*)$ if sample $S$ does not contain any points sampled from $(b^*, c^*)$.




Denote $L_{FP}$, $L_{FN,1}$, $L_{FN,2}$ these type of errors, where:

$$\begin{aligned}
L_{FP}(h_S) &= \underset{x\sim\mathcal{D}}{P}(x \in [a_S, b_S] \cup [c_S, d_S] \setminus ([a^*, b^*] \cup [c^*, d^*])) \\
&= \underset{x\sim\mathcal{D}}{P}(x \in [b^*, c^*] \subseteq [a_S, b_S] \cup [c_S, d_S]) \\
L_{FN,1}(h_S) &= \underset{x\sim\mathcal{D}}{P}(x \in [a^*, b^*] \setminus [a_S, b_S]) \\
L_{FN,2}(h_S) &= \underset{x\sim\mathcal{D}}{P}(x \in [c^*, d^*] \setminus [c_S, d_S])
\end{aligned}$$

So, if we want to have $L_{\mathcal{D},h^*}(h_S) > \epsilon$, then one of the numbers $L_{FP}$, $L_{FN,1}$, $L_{FN,2}$ must be $> \frac{\epsilon}{3}$.

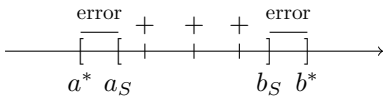
Define

$$\begin{aligned}
F_1 &= \left\{ S \sim \mathcal{D}^m \mid L_{FP}(h_S) > \frac{\epsilon}{3} \right\} \\
F_2 &= \left\{ S \sim \mathcal{D}^m \mid L_{FN,1}(h_S) > \frac{\epsilon}{3} \right\} \\
F_3 &= \left\{ S \sim \mathcal{D}^m \mid L_{FN,2}(h_S) > \frac{\epsilon}{3} \right\}
\end{aligned}$$

So,

$$\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S)\geq\epsilon)\leq\underset{S\sim\mathcal{D}^m}{P}(F_1\cup F_2\cup F_3)\leq\sum_{i=1}^{3}P(F_i)$$

$$\begin{aligned}P(F_1)&=\underset{S\sim\mathcal{D}^m}{P}\left(L_{FP}(h_S)>\frac{\epsilon}{3}\right)\\&=\left(\text{this means that }\mathcal{D}([b^*,c^*])>\frac{\epsilon}{3}\text{ and no point from }[b^*,c^*]\text{ is sampled in }S\right)\\&\leq\left(1-\frac{\epsilon}{3}\right)^m\leq e^{-\frac{\epsilon}{3}m}\\P(F_2)&=\underset{S\sim\mathcal{D}^m}{P}\left(L_{FN,1}(h_S)>\frac{\epsilon}{3}\right)\end{aligned}$$




Construct $R_1=[a^*,a_0]$ and $R_2=[b_0,b^*]$ such that $\mathcal{D}(R_1)=\mathcal{D}(R_2)=\frac{\epsilon}{6}$.

If $[a_S,b_S]\cap R_1\neq\varnothing$ and $[a_S,b_S]\cap R_2\neq\varnothing$, then the error made by $h_S$ on $[a^*,b^*]$ is smaller than $\frac{\epsilon}{6}+\frac{\epsilon}{6}\geq\frac{\epsilon}{3}$.

So $L_{FN,1}(h_S)>\frac{\epsilon}{3}\Rightarrow[a_S,b_S]\cap R_1=\varnothing$ or $[a_S,b_S]\cap R_2=\varnothing$.

Define

$$\begin{aligned}F_{21}&=\{S\sim\mathcal{D}^m\mid[a_S,b_S]\cap R_1=\varnothing\}\\F_{22}&=\{S\sim\mathcal{D}^m\mid[a_S,b_S]\cap R_2=\varnothing\}\end{aligned}$$

$$P(F_2)\leq P(F_{21}\cup F_{22})\leq P(F_{21})+P(F_{22})=2\cdot\left(1-\frac{\epsilon}{6}\right)^m\leq 2\cdot e^{-\frac{\epsilon}{6}m}$$

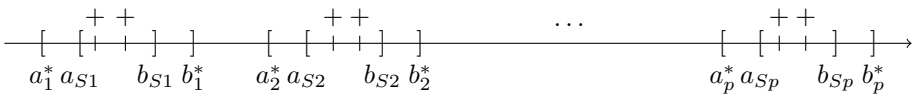
In the same way we can prove that $P(F_3)\leq 2\cdot e^{-\frac{\epsilon}{6}m}$. So we obtain that

$$\underset{S\sim\mathcal{D}^m}{P}(L_{\mathcal{D},h^*}(h_S)>\epsilon)\leq e^{-\frac{\epsilon}{3}m}+4\cdot e^{-\frac{\epsilon}{6}m}\leq e^{-\frac{\epsilon}{6}m}+4\cdot e^{-\frac{\epsilon}{6}m}=5\cdot e^{-\frac{\epsilon}{6}m}<\delta$$

$$\begin{aligned}&\Rightarrow e^{-\frac{\epsilon}{6}m}<\frac{\delta}{5}\quad\Big|\cdot\log\\&\Rightarrow-\frac{\epsilon}{6}m<\log\frac{\delta}{5}\quad\Big|\cdot\left(-\frac{6}{\epsilon}\right)\\&\boxed{m>\frac{6}{\epsilon}\cdot\log\frac{5}{\delta}}\end{aligned}$$

In the general case, for $\mathcal{C}_p=$ reunion of $p$ intervals, the proof is similar, the only differences are that:

- there are $(p-1)$ regions of false positives




- $2p$ regions of false negatives

So we have

$$\boxed{m \geq \frac{2(2p-1)}{\epsilon} \cdot \log \frac{p+2p-1}{\delta}}$$

$$\boxed{m \geq \frac{2(2p-1)}{\epsilon} \cdot \log \frac{3p-1}{\delta}}$$

time complexity $\rightarrow$ given by sorting $S = \mathcal{O}(m \log m)$

**Exercise 8.** Let $\chi$ be a domain and let $D_1, D_2, \ldots, D_m$ be a sequence of distributions over $\chi$. Let H be a finite class of binary classifiers over $\chi$ and let $f \in H$. Suppose we are getting a sample S of m examples, such that the instances are independent but are not identically distributed; the $i^{th}$ instance is sampled from $D_i$ and then $y_i$ is set to be $f(x_i)$. Let $\bar{D}_m$ denote the average, that is, $\bar{D}_m = \frac{D_1 + \cdots + D_m}{m}$. Fix an accuracy parameter $\epsilon \in (0,1)$. Show that

$$\mathbb{P}[\exists h \in H s.t. L_{(\bar{D}_m, f)}(h) > \epsilon and L_{(S,f)=0}] \leq |H| e^{-}.$$

**Solution.**
Let $h \in \mathcal{H}$, with $L_{\overline{D}_m, \delta}(h) > \epsilon \Rightarrow$

$$\underset{x \sim \overline{D}_m}{P}(h(x) \neq f(x)) > \epsilon \Leftrightarrow \underset{x \sim \overline{D}_m}{P}(h(x) = f(x)) = 1 - \underset{x \sim \overline{D}_m}{P}(h(x) \neq f(x)) < 1 - \epsilon$$

$$\begin{aligned}
\underset{x \sim \overline{D}_m}{P}(h(x) = f(x)) &= \left( x \text{ can be sampled from each } D_i, \text{ with probability } \frac{1}{m} \right) \\
&= \frac{1}{m} \cdot \underset{x \sim \overline{D}_1}{P}[h(x) = f(x)] + \cdots + \frac{1}{m} \cdot \underset{x \sim \overline{D}_m}{P}[h(x) = f(x)] \\
&= \frac{1}{m} \sum_{i=1}^{m} \underset{x \sim \overline{D}_i}{P}[h(x) = f(x)] < 1 - \epsilon
\end{aligned}$$

Consider the training set $S = \{(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_m, f(x_m)) \mid \text{where } \; x_i \sim D_i\}$

$h$ consistent with $S$ if $L_S(h) = 0$.

$$
\begin{aligned}
\underset{S\sim D_1\times D_2\times\cdots\times D_m}{P}[L_S(h)=0] &= \prod_{i=1}^{m} \underset{x_i\sim D_i}{P}[h(x_i)=f(x_i)] \\
&= \prod_{i=1}^{m} \underset{x\sim D_i}{P}[h(x)=f(x)] \\
&= \left[\left(\prod_{i=1}^{m} \underset{x\sim D_i}{P}[h(x)=f(x)]\right)^{\frac{1}{m}}\right]^m \\
&= \text{geometric mean } = (a_1\cdot a_2\cdot\cdots\cdot a_m)^{\frac{1}{m}} \\
&\quad \text{where } a_i = \underset{x\sim D_i}{P}[h(x)=f(x)] = \begin{array}{c}\text{probability that } h \text{ correctly}\\ \text{labels a point } x\sim D_i\end{array} \\
&\leq \text{arithmetic mean} = \left(\frac{a_1+a_2+\cdots+a_m}{m}\right) \\
&\leq \left[\frac{1}{m}\sum_{i=1}^{m} \underset{x\sim D_i}{P}[h(x)=f(x)]\right]^m < (1-\epsilon)^m \leq e^{-\epsilon m}
\end{aligned}
$$

There are at most $|\mathcal{H}|$ number of $h$ hypotheses. So, we observe that

$$
P\left[\exists h\in\mathcal{H} \text{ s.t. } L_{(\overline{D}_m,f)}(h)>\epsilon \text{ and } L_{(S,f)}(h)=0\right] \leq |\mathcal{H}|\cdot e^{-\epsilon m}
$$

# 2 VC-dimension

**Exercise 1.** Let H = $\{[\theta, \infty)|\theta \in \mathbb{R}\} \cup \{(-\infty, \theta)|\theta \in \mathbb{R}\}$. Compute VCdim(H).

**Solution.**

H = $\{[\theta, \infty)|\theta \in \mathbb{R}\} \cup \{(-\infty, \theta)|\theta \in \mathbb{R}\}$

Let $H_1$ and $H_2$ be the first and the last part of the union, respectively.

In other words, H is the set of all half-closed intervals, with the mention that right-open intervals are open at both sides.

We first show that VCdim(H) $\geq 2$. For this we consider $x_1, x_2 \in \mathbb{R}, x_1 < x_2$. We have the following possible labellings:

$H(x_1) = H(x_2) = 0$, pick either $\theta > x_2$(for $H_1$) or $\theta \leq x_1$(for $H_2$).

$H(x_1) = H(x_2) = 1$, pick either $\theta > x_2$(for $H_2$) or $\theta \leq x_1$(for $H_1$).

$H(x_1) = 0, H(x_2) = 1$, pick $\theta \in (x_1, x_2)$ (for $H_2$).

$H(x_1) = 1, H(x_2) = 0$, pick $\theta \in (x_1, x_2]$ (for $H_1$).

Thus VCdim(H) $\geq 2$. Is VCdim(H) $< 3$?

Let $x_1, x_2, x_3 \in \mathbb{R}, x_1 < x_2 < x_3$ and $H(x_1) = 0, H(x_2) = 1, H(x_3) = 0$.

No half-open interval from $H_1$ would be able to classify the first two points without misclassifying the last and no half-open interval from $H_2$ would be able to classify the last two correctly without miscalssifying the first point.

So VCdim(H) = 2.

**Exercise 2.** Consider H to be the class of all centered in origin sphere classifiers in the 3D space. A centered in origin sphere classifier in the 3D space is a classifier $h_r$ that assigns the value 1 to a point if and only if it is inside the sphere with radius $r > 0$ and center given by the origin O(0,0,0). Consider the realizability assumption. Compute VCdim(H).

**Solution.**

It's easy to see that VCdim(H) $\geq 1$. Let $x \in \mathbb{R}^3$. If we want h(x) = 1, then we need $r \geq ||x||$. Otherwise, we will choose $r < ||x||$.

Now let us prove that VCdim(H) $< 2$. Let $x_1, x_2 \in \mathbb{R}^3, 0 < ||x_1|| < ||x_2||$. In this case it's impossible to obtain the labelling (0, 1). In order for $h(x_2) = 1$, we would need $r \geq ||x_2||$, but that would misclassify $x_1$.

So VCdim(H) = 1.

**Exercise 3.** Let H = $\{[\theta, \theta + 1] \cup [\theta + 2, \infty)|\theta \in \mathbb{R}\}$. Compute VCdim(H).

**Solution.**

We first prove that VCdim(H) $\geq 2$. Let $x_1, x_2 \in \mathbb{R}, x_1 < x_2, x_2 - x_1 = 1 - \epsilon, \epsilon \simeq 0$. We consider the possible labellings:

$H(x_1) = H(x_2) = 0$, pick $\theta > x_2$.

$H(x_1) = H(x_2) = 1$, pick $\theta < x_1 - 2$.

$H(x_1) = 0, H(x_2) = 1$, pick $\theta = x_2$.

$H(x_1) = 1, H(x_2) = 0$, pick $\theta = x_1 - 1$.

H shatters $\{x_1, x_2\}$, thus VCdim(H) $\geq 2$.

We now check whether VCdim(H) $\geq 3$. Let $x_1, x_2, x_3 \in \mathbb{R}, x_1 < x_2 < x_3$ and $x_2 - x_1 = x_3 - x_2 = 1 - \epsilon, \epsilon \simeq 0$. Consider the following labellings:

$H(x_1) = H(x_2) = H(x_3) = 0$, pick $\theta > x_3$.

$H(x_1) = H(x_2) = H(x_3) = 1$, pick $\theta < x_1 - 2$.

$H(x_1) = 1, H(x_2) = 1, H(x_3) = 0$, pick $\theta = x_1$.

$H(x_1) = 1, H(x_2) = 0, H(x_3) = 0$, pick $\theta = x_1 - \frac{3}{2}\epsilon$.

$H(x_1) = 0, H(x_2) = 1, H(x_3) = 1$, pick $\theta = x_2 - 2$.

$H(x_1) = 0, H(x_2) = 1, H(x_3) = 0$, pick $\theta = x_1 + \epsilon$.

$H(x_1) = 0, H(x_2) = 0, H(x_3) = 1$, pick $\theta = x_3$.

So VCdim(H) $\geq 3$.

We continue to prove that VCdim(H) $< 4$.

Let $x_1, x_2, x_3, x_4 \in \mathbb{R}, x_1 < x_2 < x_3 < x_4$. We consider the labelling $H(x_1) = 1, H(x_2) = 0, H(x_3) = 1, H(x_4) = 0$. It is impossible to label the first three points properly without mislabelling the last point, since it will fall in $[\theta, \infty)$.

This proves that VCdim(H) $< 4$.

So VCdim(H) $= 3$.

**Exercise 4.** An axis aligned square classifier in the plane is a classifier that assigns the value 1 to a point if and only if it is inside a certain square. Formally, given the real numbers $a_1, a_2, r > 0 \in R$ we define the classifier $h(a_1, a_2, r)$ by:

$$H_{a_1,a_2,r}(x_1, x_2) = \begin{cases} 1, \text{if } a_1 \leq x_1 \leq a_1 + r, a_2 \leq x_2 \leq a_2 + r \\ 0, \text{otherwise} \end{cases}$$

Compute VCdim(H).

**Solution.**

To show that VCdim(H) $\geq 2$ is simple. For this, we need only consider two points in $\mathbb{R}^2$, set one of our square's vertices in one of these points and enlarge or shrink it so it contains (or not) the other point.

We now move on to showing that VCdim(H) $\geq 3$. Let $A(1,0), B(0,1), C(0,-1) \in \mathbb{R}^2$. Consider the following labellings:

$H(A) = H(B) = H(C) = 1$, pick a square that contains all points.

$H(A) = H(B) = H(C) = 0$, pick a square that does not contain any point.

For cases in which only one point is labeled with 1, we define the minimal area square that contains the 1-labelled point.

For cases in which two points are labelled with 1, we define the square that contains those two points as vertexes and does not contain the last point.

H shatters $\{A, B, C\}$, so VCdim(H) $\geq 3$. Let us study if VCdim(H) $< 4$.

Let $A, B, C, D \in \mathbb{R}^2$ with $AC > BD$ (such as in figure 3). The labelling $H(A) = 1, H(B) = 0, H(C) = 1, H(D) = 0$ is impossible, since any square that labels A and C correctly will mislabel B and D (the same goes for any square that properly classifies B and D).

This proves that the VCdim(H) $< 4$.

So VCdim(H) $= 3$.

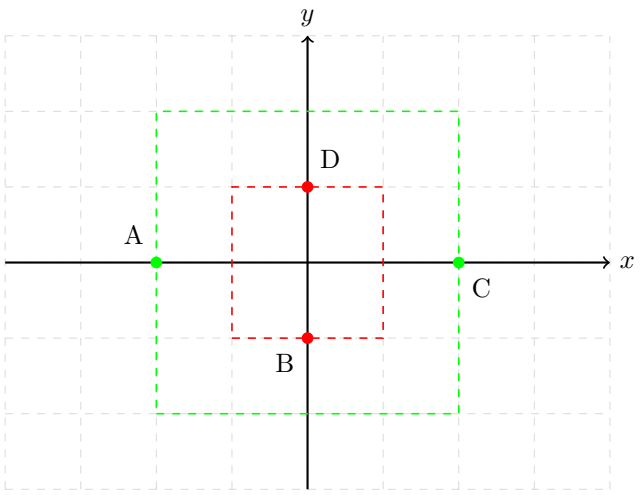



Figure 2: Unshatterable example. A square that would classify A and C correctly would also contain B and D, thus misclassifying them, while a square that would not contain B and D would misclassify A and C.

**Exercise 5.** Consider the setting of binary classification (with zero-one loss) and the following hypothesis class:

$$H = \{f : \mathbb{R} \to \{0, 1\} \mid f(x) = \mathbf{1}\{x \in [-a, a] \cup [b, \infty)\}, 0 \leq a \leq b \in \mathbb{R}\}.$$

In other words, the feature space $\chi$ is the real line, and this is the class of indicator sets of the form $x \in [-a, a] \cup [b, \infty)$ with $0 \leq a \leq b \in \mathbb{R}$.

What is VCdim(H)?

**Solution.**

Can three points be shattered? This is not immediately clear, but actually three points cannot be shattered. Let $x_1 \leq x_2 \leq x_3$ be three arbitraty points. Consider first the case $|x_1| \leq |x_2|$ and suppose $\exists h \in H$ achieving the lagelling (0, 1, 0).

It is clearly necessary to use an element of H parametrized by a nd b so that $b > x_3$. Furthermore necessarily $a \geq |x_2|$ (so that $x_2 \in [-a, a]$). However, this means that $a \geq |x_1|$ as well, meaning h induces the labeling (1, 1, 0), so we have a contradiction.

A similar situation happens when $|x_1| > |x_2|$ and we consider the labeling (1, 0, 0), which means three points cannot be shattered.

To see that two points can be shattered we can give a concrete example. let $x_1 = -1, x_2 = 2$. Then the following cases occur:

(0,0) a = 0, b = 3 $\rightarrow [0, 0] \cup [3, \infty)$.

(0,1) a = 0, b = 1 $\rightarrow [0, 0] \cup [1, \infty)$.

(1,0) a = 1, b = 3 $\rightarrow [-1, 1] \cup [3, \infty)$.

(1,1) a = 1, b = 1 $\rightarrow [-1, 1] \cup [1, \infty)$.

**Exercise 6.** Consider the setting of binary classification (with zero-one loss) and the following hypothesis class:

$$H = \{f : \mathbb{R} \rightarrow \{0, 1\} | f(x) = \mathbf{1}\{x \in [a, b] \cup [c, \infty)\}, 0 \leq a \leq b \leq c \in \mathbb{R}\}.$$

In other words, the feature space $\chi$ is the real line, and this is the class of indicator sets of the form $x \in [a, b] \cup [c, \infty)$ with $0 \leq a \leq b \leq c \in \mathbb{R}$. What is VCdim(H)?

**Solution.**

Clearly four points cannot be shattered. Let $x_1 \leq x_2 \leq x_3 \leq x_4$ be arbitrarily chosen four points on the real line. Then the labeling (1, 0, 1, 0) cannot be obtained by any element in H. Suppose this is not the case. Then necessarily there is an element in H parameterized by a,b and c such that $c \geq x_4, b \geq x_3$ and $a \leq x_1$. But in that case $a \leq x_2 \leq b$, meaning $x_2$ would be labelled 1, which is a contradiction.

To see that three points can be shattered we can give a concrete example. Let $x_1 = -1, x_2 = 1$ and $x_3 = 3$. Then the following cases occur:

(0, 0, 0) (a = 0, b = 0, c = 4) $\rightarrow [0, 0] \cup [3, \infty)$

(0, 0, 1) (a = 0, b = 0, c = 2) $\rightarrow [0, 0] \cup [1, \infty)$

(0, 1, 0) (a = 0, b = 1, c = 4) $\rightarrow [-1, 1] \cup [3, \infty)$

(0, 1, 1) (a = 0, b = 1, c = 2) $\rightarrow [-1, 1] \cup [1, \infty)$

(1, 0, 0) (a = -1, b = 0, c = 4) $\rightarrow [0, 0] \cup [3, \infty)$

(1, 0, 1) (a = -1, b = 0, c = 2) $\rightarrow [0, 0] \cup [1, \infty)$

(1, 1, 0) (a = -1, b = 1, c = 4) $\rightarrow [-1, 1] \cup [3, \infty)$

(1, 1, 1) (a = -1, b = 1, c = 2) $\rightarrow [-1, 1] \cup [1, \infty)$

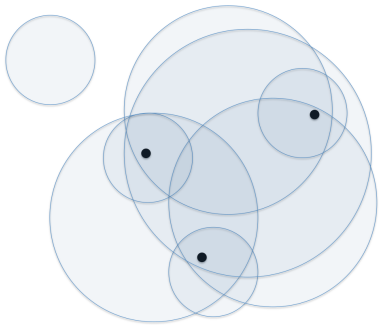
**Exercise 7.** Consider the class of circles in the plane, that is:

$$H = \{h \mid h(x,y) = \mathbf{1}\{(x-a)^2 + (y-b)^2 \leq r^2\}\ a, b, r \in \mathbb{R}\}$$

What is VCdim(H)?

**Solution.**

It is easy to see that 1, 2 and 3 points can be shattered by circles in the plane (provided they are not collinear). An example of three points in the plane, and the $8 = 2^3$ circles shattering them (i.e. elements from the class achieving all possible labelings) is given in the figure below.



If we take n = 4, things start to get complicated. WE consider two cases:

- If one of the 4 points, say $x_1$, lies within the convex hull of the other 3 points, then the labelling (0, 1, 1, 1) cannot be achieved.
- Otherwise, label the points in clockwise order $x_1, x_2, x_3, x_4$. Note that if there was a circle $c_1$ that achieves the label sequence 1, 0, 1, 0 and another circle $c_2$ that achieves the label sequence 0, 1, 0, 1 then their symmetric difference $c_1 \setminus c_2$ should consist of (at least) 4 disjoint regions. However, this is impossible for circles.

**Exercise 8.** Given some domain set, $\chi$ and a number $k \leq |\chi|$, compute the VC-dimension of the following hypothesis classes:

a) $H^{\chi}_{=k} = \{h \in \{0,1\}^k \mid |\{x|h(x) = 1\}| = k\}$, that is, the set of all functions that assign the value 1 to at most k elements of $\chi$.

b) $H^{\chi}_{\leq k} = \{h \in \{0,1\}^k \mid |\{x|h(x) = 1\}| \leq k \text{ or } |\{x|h(x) = 0\}| \leq k\}$

**Solution.**

We will refer to the hypothesis classes as simply H in both subsections.

a) We claim that VCdim(H) = min{k, $|\chi|$ - k}.

First, we show that VCdim(H) $\leq k$. Let $C \subset \chi$ be a set of size K + 1. Then, there doesn‘t exist $h \in H$ that satisfies $h(x) = 1$ for all $x \in C$.

Analogously, if $C \subset \chi$ is a set of size $|\chi| - k + 1$, there is no $h \in H$ which satisfies h(x) = 0 for all $x \in C$. Hence, VCdim(H) $\leq \min\{k, |\chi| - k\}$.

It's left to show that VCdim(H) $\geq \min\{k, |\chi| - k\}$. Let C = $\{x_1, ..., x_m\} \subset \chi$ be a set with size m $\geq \min\{k, |\chi| - k\}$. Let $\{x_1, ..., x_m\} \subset \{0,1\}^m$ be a vector of labels. Denote $\sum_{i=1}^{m} y_i = s$. Pick an arbitrarily subset $E \in \chi \setminus C$ of k - s elements and let $h \in H$ be a hypothesis which satisfies $h(x_i) = y_i, \forall x_i \in C$ and $h(x) = \mathbf{1}_{[E]}, \forall x \in \chi \setminus C$. We conclude that C is shattered by H. It follows that VCdim(H) $\geq \min\{k, |\chi| - k\}$.

b) We claim that VCdim(H) = k.

First, we show that VCdim(H) $\leq k$. Let $C \subset \chi$ be a set of size k + 1. Then, $\nexists\, h \in H$ that satisfies h(x) = $1, \forall x \in C$.

It's left to show that Vcdim(H) *geq* k. Let C = $\{x_1, ..., x_m\} \subset \chi$ be a set of size $m \leq k$. Let $\{x_1, ..., x_m\} \subset \{0,1\}^m$ be a vector of labels. This labeling is obtained by some $h \in H$ which satisfies $h(x_i) = y_i, \forall x_i \in C$ and $h(x) = 0, \forall x \in \chi \setminus C$. We conclude that C is shattered by H. It follows that VCdim(H) $\geq k$.

**Exercise 9.** Consider the class $\mathcal{H}^d_{mcon}$ of monotone Boolean conjunctions over $\{0,1\}^d$. Monotonicity here means that the conjunctions do not contain negations. As in $\mathcal{H}^d_{mcon}$, the empty conjunction is interpreted as the all-positive hypothesis. We augment $\mathcal{H}^d_{mcon}$ with the all-negative hypothesis $h^-$. Show that VCdim($\mathcal{H}^d_{mcon}$) = $d$.

**Solution.**

$\mathcal{H}^d_{\text{mcon}}$ is the class of monotone Boolean conjunctions over $\{0,1\}^d$.

$$\mathcal{H}^d_{\text{mcon}} = \left\{ h\colon \{0,1\}^d \to \{0,1\},\ h_{(x_1,x_2,\ldots,x_d)} = \bigwedge_{i=1}^{d} l(x_i) \right\} \cup \{ \underset{\substack{\downarrow \\ h^-(x_1,\ldots,x_d)=0 \\ \text{always}}}{h^-} \}$$

$$l(x_i) \in \{ \underset{\substack{\downarrow \\ \text{positive} \\ \text{literal}}}{x_i}, \underset{\substack{\downarrow \\ \text{missing} \\ \text{literal}}}{1} \}$$

So $|\mathcal{H}^d_{\text{mcon}}| = 2^d + 1$.

Examples:

$d = 2 \qquad \mathcal{H}^2_{\text{mcon}} = \{ \underset{\substack{\downarrow \\ h^-}}{0}, \underset{\substack{\downarrow \\ h_{\text{empty}}}}{1}, x_1, x_2, x_1 \wedge x_2 \}$

$d = 3 \qquad \mathcal{H}^3_{\text{mcon}} = \{0, 1, x_1, x_2, x_3, x_1 \wedge x_2, x_1 \wedge x_3, x_2 \wedge x_3, x_1 \wedge x_2 \wedge x_3\}$

We need to show that $VC\dim \mathcal{H}^d_{\text{mcon}} = d$.

We know that $|\mathcal{H}^d_{\text{mcon}}| = 2^d + 1$, so $VC\dim \mathcal{H}^d_{\text{mcon}} \leq \lfloor \log_2(|\mathcal{H}^d_{\text{mcon}}|) \rfloor$, which in turn means

$$VC\dim \mathcal{H}^d_{\text{mcon}} \leq \lfloor \log_2(2^d + 1) \rfloor = d$$

We only need to find a set $C \subseteq \{0,1\}^d$ with $d$ points that is shattered by $\mathcal{H}^d_{\text{mcon}}$.

Usually, taking $C = \{e_1, e_2, \ldots, e_d\}, e_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ works, but not for this $\mathcal{H} = \mathcal{H}^d_{\text{mcon}}$. You cannot have a conjunction that will have $h(e_1) = h(e_2) = 1$ and $h(e_3) = \cdots = h(e_d) = 0$.

Instead, we choose $C = \{(0,1,1,\ldots,1), (1,0,1,1,\ldots,1), \ldots, (1,1,\ldots,\underset{i}{0},1,1)\}$ set of vectors of the form $c_i = (1,1,\ldots,1) - e_i, i = \overline{1,d}$.

We want to show that, for each possible labeling $(y_1, y_2, \ldots, y_d)$ of the points $c_i = (1,1,\ldots,1) - e_i$, there exists a function $h \in \mathcal{H}^d_{\text{mcon}}$ such that $h(c_i) = y_i, \forall i = \overline{1,d}$.

Consider $(y_1, y_2, \ldots, y_d)$ a labeling and take $\mathcal{J} = \{j \mid y_j = 1\}$.

If $\mathcal{J} = \varnothing$, then $h^-$ realizes the labeling $(0,0,\ldots,0)$.

If $\mathcal{J} = \{1,2,\ldots,d\}$, then $h_{\text{empty}} = 1$ (all literals are missing) realizes the labeling $(1,1,\ldots,1)$.

If $1 \leq |\mathcal{J}| \leq d-1$, then consider $h_{\mathcal{J}}(x_1, x_2, \ldots, x_d) = \bigwedge\limits_{j \notin \mathcal{J}} x_j$.

For example, if $d = 4$ and $\mathcal{J} = \{2,3\}$, $h_{\mathcal{J}}(x_1, x_2, x_3, x_4) = x_1 \wedge x_4$:

$$h_{\mathcal{J}}(c_1) = h_{\mathcal{J}}(0,1,1,1) = 0$$
$$h_{\mathcal{J}}(c_2) = h_{\mathcal{J}}(1,0,1,1) = 1$$
$$h_{\mathcal{J}}(c_3) = h_{\mathcal{J}}(1,1,0,1) = 1$$
$$h_{\mathcal{J}}(c_4) = h_{\mathcal{J}}(1,1,1,0) = 0$$

We have that $h_{\mathcal{J}}(c_i) = 1$ if $i \in \mathcal{J}$ and $h_{\mathcal{J}}(c_i) = 0$ if $i \notin \mathcal{J}$.

For all indices $i \in \mathcal{J}$, $c_i$ will have value 0 on the position $i$ and 1 in rest, but variable $x_i$ is not considered in the conjunction. So $h_{\mathcal{J}}(c_i) = 1$.

For all indices $i \notin \mathcal{J}$, $c_i$ will have value 0 on the position $i$ and, because the conjunction contains the literal $x_i$, then we have that $h_{\mathcal{J}}(c_i) = 0$.

**Exercise 10.** Let $\mathcal{X}$ be the Boolean hypercube $\{0,1\}^n$. For a set $I \subseteq \{1,2,\ldots,n\}$ we define a *parity function* $h_I$ as follows. On a binary vector $x = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$,

$$h_I(x) = \left( \sum_{i \in I} x_i \bmod 2 \right)$$

(That is, $h_I$ computes parity bits in $I$.) What is the VC-dimension of the class of all such parity functions, $\mathcal{H}_{n-parity} = \{h_I \mid I \in \{1,2,\ldots,n\}\}$?

**Solution.**

$\mathcal{X} = \{0,1\}^n$

$$\mathcal{H}_{\text{n-parity}} = \left\{ h_I \mid I \subseteq \{1,2,\ldots,n\},\ h_I(x_1, x_2, \ldots, x_n) = \left( \sum_{i \in I} x_i \right) \bmod 2 \right\}$$

What is $VC\dim \mathcal{H}_{\text{n-parity}}$?

For each subset $I \subseteq \{1, 2, \ldots, n\}$ we have a $h_I$, so $|\mathcal{H}_{\text{n-parity}}| = 2^n$.
We know that $VC\dim \mathcal{H}_{\text{n-parity}} \leq \lfloor \log_2 2^n \rfloor = n$.
So $VC\dim \mathcal{H}_{\text{n-parity}} \leq n$.
Can we find a set $C$ with $n$ points that is shattered by $\mathcal{H}_{\text{n-parity}}$?
Let's try the "usual" set of unit vectors $C = \{e_1, e_2, \ldots, e_n\}, e_i = (0, \ldots, 0, \underset{i}{1}, 0, \ldots, 0)$.

We want to show that, for each possible labeling $(y_1, y_2, \ldots, y_n)$ of $(e_1, e_2, \ldots, e_n)$, you can find a corresponding $h$ such that $h(e_i) = y_i, \forall i = \overline{1, n}$.

Consider $(y_1, y_2, \ldots, y_n)$ such a labeling and take $I = \{i \mid y_i = 1\}$.
Then we have

$$h_I(e_i) = \left(\sum_{i \in I} x_i\right) \mod 2 = \begin{cases} 1, & \text{if } i \in I \\ 0, & \text{otherwise} \end{cases}$$

So $VC\dim \mathcal{H}_{\text{n-parity}} = n$.

**Exercise 11.** Given some finite domain ser, $\mathcal{X}$, and a number $k \leq |\mathcal{X}|$, figure out the VC-dimension of each of the following classes (and prove your claims):

1. $\mathcal{H}^{\mathcal{X}_{=k}} = \{h \in \{0,1\}^{(X)} \mid |\{x \mid h(x) = 1\} = k|\}$, that is, the set of all functions that assign the value of 1 to exactly $k$ elements of $\mathcal{X}$.
2. $\mathcal{H}^{\mathcal{X}_{=k}} = \{h \in \{0,1\}^{(X)} \mid |\{x \mid h(x) = 1\} \leq k| \text{ or } |\{x \mid h(x) = 0\} \leq k|\}$

**Solution.**
$\mathcal{X}$ is finite domain, $|\mathcal{X}| < \infty$, $k \leq |\mathcal{X}|$

**3.1.** $\mathcal{H}^{\mathcal{X}}_{=k} = \left\{h \in \{0,1\}^{\mathcal{X}} \,\middle|\, |\{x \colon h(x) = 1\}| = k\right\}$
= set of all functions that assign the value 1 to exactly $k$ elements of $\mathcal{X}$

$VC\dim \mathcal{H}^{\mathcal{X}}_{=k} = ?$

If $k = 0 \Rightarrow \mathcal{H}^{\mathcal{X}}_{=0} = \{h^-\}$, all points get the value 0
If $k = 1 \Rightarrow \mathcal{H}^{\mathcal{X}}_{=1}$ has $|\mathcal{X}|$ functions $= n$ functions
$\mathcal{X} = \{x_1, x_2, \ldots, x_n\}, n = |\mathcal{X}|$

$$h_i \colon \{x_1, \ldots, x_n\} \to \{0,1\} \quad h_i(x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

If $k = 2 \Rightarrow \mathcal{H}^{\mathcal{X}}_{=2}$ has $C^2_n$ elements

$\vdots$

If $k = n - 1 \Rightarrow \mathcal{H}^{\mathcal{X}}_{=n-1}$ has $n$ elements
If $k = n \Rightarrow \mathcal{H}^{\mathcal{X}}_{=n}$ has 1 element $h^+(x_i) = 1 \; \forall i = \overline{1, n}$

We first show that $VC\dim \mathcal{H}^{\mathcal{X}}_{=k} \leq \min(k, n-k)$.

$$VC\dim \mathcal{H}^{\mathcal{X}}_{=k} \;\|\; VC\dim \mathcal{H}$$

Case 1) if $n \geq 2k$, in this case, $\min(k, n-k) = k,\ k \leq \frac{n}{2}$

$\mathcal{H}$ will consist of functions $h$ that label exactly $k$ elements of $\mathcal{X}$ with label 1. So any set $C$ with more than $k$ points cannot be shattered because the labeling with all 1's $(1, 1, 1, \ldots, 1)$ cannot be realized by any $h \in \mathcal{H}$.

Case 2) if $n < 2k$, in this case $\min(k, n-k) = n-k,\ k > \frac{n}{2}$

$\mathcal{H}$ will consist of functions $h$ that labels $k$ elements of $\mathcal{X}$ with label 1, and $n-k$ points of $\mathcal{X}$ with label 0. So any set with more than $n-k+1$ points cannot be shattered by $\mathcal{H}$ as the labeling with all 0's $(0, 0, \ldots, 0)$ cannot be realized by any $h \in \mathcal{H}$.

So we have that $VC\dim \mathcal{H} \leq \min(k, n-k)$.

We will prove that $VC\dim \mathcal{H} = \min(k, n-k)$.

Consider $k' = \min(k, n-k)$.

We need to show that there exists a set of points $A = \{x_{i1}, x_{i2}, \ldots, x_{ik}$ $\} \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$. This means that, for each subset $B \subseteq A$, we can find $h_B \in \mathcal{H}$ such that

$$h_B(x) = \begin{cases} 1, & x \in B \\ 0, & x \in A \setminus B \end{cases}$$

We choose a set of $k - |B|$ points $B' = \{b_1, b_2, \ldots, b_{k-|B|}\} \subseteq \mathcal{X} \setminus A$.

We can make the choice since $k - |B| \leq |\mathcal{X} \setminus A|$

$$k - |B| \leq n - k$$
$$k' - |B| \leq n - k$$
$$k' - |B| \leq k' \leq n - k \text{ (this is true)}$$

So $B \subseteq A$ has $|B|$ elements

$B \subseteq \mathcal{X} \setminus A$ has $k - |B|$ elements

So $|B \cup B'| = |B| + |B'|$ (as $B \cap B' = \varnothing) = k$

So, if we consider the characteristic function of the set $B \cup B'$, we have

$$\mathbb{1}_{B \cup B}$$

$$(\mathrm{x}) = \begin{cases} 1, & x \in B \cup B' \\ 0, & \text{otherwise} \end{cases}$$

What is more important, $\mathbb{1}_{B \cup B}$

$takes value 1 for exactly k points, so it is a member of \mathrm{H}$.

So, in this case, we take $h_B = \mathbb{1}_{B \cup B}$

.

$h_B$ will have the desired property that $h_B(x) = \begin{cases} 1, & x \in B \\ 0, & x \in A \setminus B \end{cases}$

So any set $A$ of $k' = min(k, n-k)$ points (?) can be shattered by $\mathcal{H}$.

So $VC\dim \mathcal{H} = k' = \min(k, n-k)$.

**3.2.** $\mathcal{H}_{\text{at-most-}k} = \left\{h \in \{0,1\}^{\mathcal{X}} \colon |\{x \colon h(x) = 1\}| \le k \text{ or } |\{x \colon h(x) = 0\}| \le k\right\}$

If $k = 0$ $\quad \mathcal{H}_{\text{at-most-0}} = \{h^-, h^+\}$ $\quad$ where $|\{x \colon h^-(x) = 1\}| \le 0$ and $|\{x \colon h^+(x) = 0\}| \le 0$

If $k = 1$ $\quad \mathcal{H}_{\text{at-most-1}} = \{h^-, h^+\} \cup \{\text{functions } h \text{ which label just one point with label 1}\}$ $\quad \hookrightarrow \mathcal{H}^{\mathcal{X}}_{=1}$

$\cup \{\text{functions } h \text{ which label just one point with label 0}\}$

Case 1 $\quad$ If $n = |\mathcal{X}| \le 2k-1$, then we have that $\mathcal{H}_{\text{at-most-}k} = \{0,1\}^{\mathcal{X}} = \{h \colon ??\mathcal{X} \to \{0,1\}\}$

This is true because any function $h \colon \mathcal{X} \to \{0,1\}$ will have either at most $k$ points $\Downarrow_{h \in \mathcal{H}_{\text{at-most-}k}}$ labeled with 0 or at most $k$ points labeled with 1.

$\Downarrow_{h \in \mathcal{H}_{\text{at-most-}k}}$

Example (see Table 1): Take $n = 7,\ k = 4$ $\quad \mathcal{X} = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$

| | $x$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ |
|---|---|---|---|---|---|---|---|---|
| at most 4 1's, 4 0's ← | $h(x)$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| at most 4 1's ← | $h(x)$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| at most 4 1's ← | $h(x)$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

Table 1

In this case, $VC\dim \mathcal{H}_{\text{at-most-}k} = VC\dim \{0,1\}^{\mathcal{X}} = n = |\mathcal{X}|$.

Case 2 $\quad$ If $n = |\mathcal{X}| \ge 2k+2$

We first show that $VC\dim \mathcal{H}_{\text{at-most-}k} = VC\dim \mathcal{H} \ge 2k+1$

$\mathcal{H}$

Consider any set $A$ of $2k+1$ points in $\mathcal{X}$: $A = \{a_1, a_2, \ldots, a_{2k+1}\}$.

We will show that $A$ is shattered by $\mathcal{H}$. It is enough to show that, for each possible labeling $(y_1, y_2, \ldots, y_{2k+1})$ of the points $(a_1, a_2, \ldots, a_{2k+1})$, we can find an $h \in \mathcal{H}$ such that $h(y_i) = a_i$.

Take $\mathcal{J} = \{j \mid y_j = 1\}$, and take $B_{\mathcal{J}} = \{a_j \in A \mid y_j = 1\}$

$j \in \mathcal{J}$

If $|\mathcal{J}| \leq k$ we know that $\mathbb{1}_{B_\mathcal{J}} \in \mathcal{H}_{\text{at-most-}k}$, so we take

$$h = \mathbb{1}_{B_\mathcal{J}} \quad h(a_j) = \begin{cases} 1, & y_j = 1 \\ 0, & \text{otherwise} \end{cases}$$

If $|\mathcal{J}| > k$ take $C_\mathcal{J} = \{a_j \in A \mid \underset{j \notin \mathcal{J}}{y_j = 0}\}$ has at most $k$ elements, so we take in this case

$$h = \mathbb{1}_{A \setminus C_\mathcal{J}} \quad h(a_j) = \begin{cases} 1, & y_j = 1 \\ 0, & y_j = 0 \end{cases}$$

So, we have that $VC\dim \mathcal{H} \geq 2k + 1$.

We show now that $VC\dim \mathcal{H} < 2k + 2$.

Consider any set $A$ of $2k + 2$ points $A = \{a_1, a_2, \ldots, a_{2k+2}\}$.

There is no $h \in \mathcal{H}$ that will label the first $k + 1$ points with 1 and the rest $k + 1$ points with 0.

So, in conclusion, $VC\dim \mathcal{H}_{\text{at-most-}k} = \min(|\mathcal{X}|, 2k + 1)$.

**Exercise 12.** Let $\Sigma$ be a finite alphabet and let $\mathcal{X} = \Sigma^m$ be the sample space of all strings of length $m$ over $\Sigma$. Let $\mathcal{H}$ be a hypothesis space over $\mathcal{X}$, where $\mathcal{H} = \{h_w : \Sigma^m \to \{0, 1\}, w \in \Sigma^*, 0 < |w| \leq m \text{ s.t. } h_w(x) = 1 \text{ if } w \text{ is a substring of } x\}$. Give any upper bound for the VC-dimension of $\mathcal{H}$ in terms of $|\Sigma|$ and $m$.

**Solution.**

Solution 1:

*Affirmation.* $VCDim(\mathcal{H}) < |\Sigma|^m$

*Proof.* Let $m \in \mathbb{N}^*$ and $\Sigma$, $\mathcal{X}$ and $\mathcal{H}$ defined as above. Assume that $VCDim(\mathcal{H}) \geq |\Sigma|^m$, then $\mathcal{H}$ should shatter a set of $|\Sigma|^m$ distinct words from $\mathcal{X}$ and the only such set is $\mathcal{X}$ itself. Take the $(\underbrace{0, 0, ..., 0}_{|\Sigma|^m})$ labeling, then there should be a $h_w \in \mathcal{H}$ such that $w$ does not exist as a substring for every word in $\mathcal{X}$.

If $m = 1$ and $|\Sigma| = 1$, assume that $\mathcal{X} = \{a\}$, then $\mathcal{H} = \{h_a\}$ since we can't have $|w| = 0$, so the (0) label is not possible.

If $m > 1$ and $|\Sigma| = 1$, assume that $\mathcal{X} = \{\underbrace{aa...a}_{m}\}$, then $\mathcal{H} = \{h_{\underbrace{aa...a}_{j}} \mid 1 \leq j \leq m\}$, so the (0) label is not possible.

In general, if $m \geq 1$ and $|\Sigma| \geq 1$ pick any $h_w \in \mathcal{H}$ and let $n$ be the length of $w$. Since $\mathcal{X}$ contains every possible word of length $m$, it will also contain the word $w + \underbrace{v...v}_{m-n}$, where $+$ is the concatenation operator and $v$ is any letter in $\Sigma$, so there will always be at least one element labeled with 1, thus the labeling $(\underbrace{0, 0, ..., 0}_{|\Sigma|^m})$ is not possible, so $\mathcal{H}$ can't shatter $\mathcal{X}$ and we get a contradiction with the assumption that $VCDim(\mathcal{H}) \geq |\Sigma|^m$. □

Solution 2:

*Affirmation.* $VCDim(\mathcal{H}) \leq log_2(|\mathcal{H}|) = log_2(|\Sigma|^1 + |\Sigma|^2 + ... + |\Sigma|^m)$

*Proof.* The class $\mathcal{H}$ has cardinality $|\mathcal{H}| = |\Sigma|^1 + |\Sigma|^2 + ... + |\Sigma|^m$ - it contains all the hypothesis $h_w$ where $w$ is a word of maximum length $m$ with elements from $\Sigma$, so $\mathcal{H}$ is a finite class. By observation 6.3.4 from [SB14], we have that $VCDim(\mathcal{H}) \leq log_2(|\mathcal{H}|) = log_2(|\Sigma|^1 + |\Sigma|^2 + ... + |\Sigma|^m)$. □

**Exercise 13. (Duality of VC dimension)** Let $\chi$ be an instance space and $H \subseteq \{0,1\}^X$ a hypothesis space with finite VC dimension. For each $x \in \chi$, we consider the function $z_x : H \to \{0,1\}$ such that $z_x(h) = h(x)$ for each $h \in H$. Let $Z = \{z_x : H \to \{0,1\} \mid x \in \chi\}$. Prove that $VCdim(Z) < 2^{VCdim(H)+1}$.

**Solution.**

We will prove that $VCdim(Z) < 2^{VCdim(H)+1}$ by contradiction.

We start by giving the following equivalent **definition**[NY15]:

*The VC dimension of a binary matrix $S$, denoted $VC(S)$, is defined as follows. A subset $C$ of columns of $S$ is called shattered if each of the $2^{|C|}$ different patterns of ones and zeroes appears in some row in the restriction of $B$ to the columns in $C$. The VC dimension of $B$ is the maximal size of a shattered subset of columns.*

Let $VCdim(H) = d$. (1)

Suppose that $VCdim(Z) \geq 2^{d+1}$.

$\exists \mathcal{Y} \subset H, |\mathcal{Y}| = 2^{d+1}$ a subset of functions and $Y \subset X, |Y| = 2^{2^{d+1}}$ a subset of points, s.t. the vectors $(z_1(x) \dots z_{|\mathcal{Y}|}(x))$ range over all possible $2^{2^{d+1}}$ binary combinations as $x$ ranges over the points in Y.

We construct the matrix $A$ by queueing all $(z_1(x) \dots z_{|\mathcal{Y}|}(x))$ vectors one after another as its columns. The first row will be comprised of $z_1$ functions, the second row will hold $z_2$ functions and so on until row $2^{d+1}$ which will hold the $z_{|\mathcal{Y}|}$ functions. The resulting matrix will be of shape $2^{d+1} \times 2^{2^{d+1}}$, with rows indexed after elements of $\mathcal{Y}$ and columns indexed after elements of $Y$.

$A$ is a row-wise representation of $\mathcal{Y}$ and a column-wise representation of $Y$.

We denote with $B$ the $2^{d+1} \times (d+1)$ matrix whose rows are binary digits of the numbers raging from 0 to $2^{d+1} - 1$. In other words, we number the rows using binary digits of length $d+1$. In particular, we have that for each $i$ there is a column that contains the $i^{th}$ bit of the row index.

For example, for $d = 2$, we have:

$$B = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$, where rows represent numbers from 0 to 7.

We know from the re-interpretation of VC dimension for binary matrices that the VC dimension of $A$ is the maximal size of a set $S$ of columns which are shattered, that is, all possible rows appear when we restrict $A$ to the columns in $S$.

Since the columns of $A$ contain all possible binary vectors of length $2^{d+1}$, then $B$ must be a submatrix of $B$. This means that $VCdim(A) = d + 1$, which is the number of columns of $B$.

Since the rows of $A$ are indexed after elements of $\mathcal{Y}$, $A$ is a row-wise binary representation of $H$. Thus, there exists a set of points of size $d+1$ which is shattered by H and therefore $VCdim(H) \geq d+1$. (2)

$(1) + (2) \Rightarrow \perp$. So $VCdim(Z) < 2^{d+1}$, with $d = VCdim(H)$.

# 3 Shattering Coefficient

**Exercise 1.** Consider the following hypothesis class:

$$H = \{h_{\theta_1} : \mathbb{R} \to \{0,1\} \mid h_{\theta_1} = \mathbf{1}_{[\theta_1,\infty)}, \theta_1 \in \mathbb{R}\} \cup \{h_{\theta_2} : \mathbb{R} \to \{0,1\} \mid h_{\theta_2} = \mathbf{1}_{(-\infty,\theta_2)}, \theta_2 \in \mathbb{R}\}$$

a) Compute the shattering coefficient $\tau_H(m)$ of the growth function for m>0.

b) Compare your result with the general upper bound for growth functions.

c) Does there exist a hypothesis class H for which $\tau_H(m)$ is equal to the general upper bound? If you answer is yes, provide an example. If your answer is no, provide a justification.

**Solution.**
a) Consider the following hypothesis classes:

$$H^1_{thresholds} = \{h_{\theta_1} : \mathbb{R} \to \{0,1\} \mid h_{\theta_1} = \mathbf{1}_{[\theta_1,\infty)}, \theta_1 \in \mathbb{R}\}$$

and

$$H^2_{thresholds} = \{h_{\theta_2} : \mathbb{R} \to \{0,1\} \mid h_{\theta_2} = \mathbf{1}_{(-\infty,\theta_2)}, \theta_2 \in \mathbb{R}\}$$

Obviously for each of these classes we have that $\tau_H(m) = m + 1$. Let's look at all the possible labelings with $H^2_{thresholds}$. Let $L^2$ be the set of all possible labelings with $H^2_{thresholds}$. We have that:

$$L^2 = \{(0,0,0,\ldots,0),(1,0,0,\ldots,0),(1,1,0,\ldots,0),\ldots,(1,1,1,\ldots,1)\}$$

It's easy to observe that $|L^2| = m + 1$. There are m labelings that contain at least one 1 label, plus $(0,0,\ldots,0)$. If we swap the order of the 1s, we get all labelings for $H^1_{thresholds}$. Let $L^1$ be the set of all possible labeling for $H^1_{thresholds}$. We have:

$$L^1 = \{(0,0,0,\ldots,0),(0,0,0,\ldots,1),(0,\ldots,0,1,1),\ldots,(1,\ldots,1,1,1)\}$$

Now, since $H = H^1_{thresholds} \cup H^2_{thresholds}$, we have $L = L^1 \cup L^2$ the set of all possible labelings of m points with H. We have that:

$$\begin{aligned} L = \{&(0,0,0,\ldots,0),(1,0,0,\ldots,0),(1,1,0,\ldots,0),\ldots,(1,1,1,\ldots,1),\\ &(0,0,0,\ldots,0),(0,0,0,\ldots,1),(0,\ldots,0,1,1),\ldots,(1,\ldots,1,1,1)\} \end{aligned}$$

The only common elements are $(0,0,\ldots,0)$ and $(1,1,\ldots,1)$, so have that:

$$\begin{aligned} \tau_H(m) &= \tau_{H^1_{thresholds}}(m) + \tau_{H^2_{thresholds}}(m) \\ &= m + 1 + m + 1 - 2 \\ &= 2m. \end{aligned}$$

b) By the Sauer-Shelah-Perles lemma, we have that $\tau_H(m) \leq \sum_{i=0}^{d} \binom{m}{i}$, where $d = VCdim(H)$.

We know from the last assignment that VCdim(H) = 2, so:

$$\begin{aligned}\tau_H(m) &\leq \sum_{i=0}^{2} \binom{m}{i} \\ &\leq \binom{m}{0} + \binom{m}{1} + \binom{m}{2} \\ &\leq \frac{m!}{0!(m-0)!} + \frac{m!}{1!(m-1)!} + \frac{m!}{2!(m-2)!} \\ &\leq 1 + m + \frac{m(m-1)}{2}.\end{aligned}$$

This shows that our result, $\tau_H(m) = m+1$, is considerably smaller than the general upper bound for growth functions.

c) Recall that $H^1_{thresholds}$ and $H^2_{thresholds}$ have shatter coefficient of $\tau_H(m) = m + 1$. Let H be one of the classes mentioned a priori. Now we set $d = VCdim(H) = 1$. By Sauer's lema we have that:

$$\tau_H(m) \leq \binom{m}{0} + \binom{m}{1} = \frac{m!}{0!(m-0)!} + \frac{m!}{1!(m-1)!} = 1 + m.$$

So the upper bound of the shatter coefficient of H is equal to $\tau_H(m)$.

**Exercise 2.** Consider the concept class H formed by closed intervals [a,b] with $a, b \in \mathbb{R}$:

$$H = \{h_{a,b} : \mathbb{R} \to \{0,1\} \mid h_{a,b} = \mathbf{1}_{[a,b]}(x), a \leq b\}$$

Compute the shattering coefficient $\tau_H(m)$ of the growth function for $m \geq 0$.

**Solution.**

We start by denoting the concept class with H so that we can use the same notations as in *Understanding Machine Learning* (Shai Ben-David, Shai Shalev-Shwartz) without causing any confusion:

$$H = \{h_{a,b} : \mathbb{R} \to \{0,1\} \mid h_{a,b} = \mathbf{1}_{[a,b]}(x), a \leq b\}$$

Let $C \subset \mathbb{R}, C = \{c_i \mid c_i \in \mathbb{R}\, c_j \neq c_k, \forall j,k, j \neq k\}, |C| = m$.

We have two cases for $H \cap C$:

1. $H \cap C = \{c_i, \ldots, c_j\}$, with $1 \leq i \leq j \leq m$.
2. $H \cap C = \varnothing$.

We will start with the second case as it is simpler. The empty set denotes that there are no 1-labeled points, so $H \cap C$ creates the labeling $\underbrace{(0,0,\ldots,0)}_{\text{m elements}}$.

Regarding the first case, there are $\binom{m+1}{2}$ possible labelings. The 1-labeled points need to be consecutive due to the nature of intervals, so we have:

$$\underbrace{\{(1,0,0,\ldots,0),(0,1,0,\ldots,0),(0,0,1,\ldots,0)\ldots(0,0,0,\ldots,1)\}}_{\text{m labelings with 1 1-labeled point.}}$$

$$\underbrace{\{(1,1,0,\ldots,0,0),(0,1,1,\ldots,0,0)\ldots(0,0,0,\ldots,1,1)\}}_{\text{m - 1 labelings with 2 1-labeled points.}}$$

$$\underbrace{\{(1,1,1,0\ldots0,0,0,0),(0,1,1,1,\ldots,0,0,0,0)\ldots(0,0,0,\ldots,0,1,1,1)\}}_{\text{m - 2 labelings with 3 1-labeled points.}}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\underbrace{\{(1,1,1,\ldots,1,0,0),(0,1,1,\ldots,1,1,0)\ldots(0,0,1,\ldots,1,1,1)\}}_{\text{3 labelings with m-2 1-labeled points.}}$$

$$\underbrace{\{(1,1,\ldots,1,0),(0,1,\ldots,1,1)\}}_{\text{2 labelings with m-1 1-labeled points.}}$$

$$\underbrace{\{(1,1,\ldots,1,1)\}}_{\text{1 labeling with m 1-labeled points.}}$$

In total we have $m+(m-1)+\ldots+1 \overset{Gauss'formula}{=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=} \frac{m\cdot(m+1)}{2} = \binom{m+1}{2}$. By adding the $(0,0,\ldots,0)$ labeling from the $2^{nd}$ case, we get $\binom{m+1}{2}+1$ labelings.

Thus, $\tau_H(m) = \binom{m+1}{2}+1$.

# 4 Boosting

**Exercise 1.** Consider the boosting algorithm described (page 4) in the article "Rapid object detection using a boosted cascade of simple features", P. Viola and M. Jones, CVPR 2001. Consider that the number of positives is equal with the number of negative examples (l = m).

a. Prove that the distribution $w_{t+1}$ obtained at round t + 1 based on the algorithm described in the article is the same with the distribution $D_{(t+1)}$ obtained based on the procedure described in the following figure:

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where $m$ and $l$ are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that $w_t$ is a probability distribution.
  2. For each feature, $j$, train a classifier $h_j$ which is restricted to using a single feature. The error is evaluated with respect to $w_t$, $\epsilon_j = \sum_i w_i \, |h_j(x_i) - y_i|$.
  3. Choose the classifier, $h_t$, with the lowest error $\epsilon_t$.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example $x_i$ is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

b. Prove that the two final classifiers (the one described in the article and the one described in the previous figure) are equivalent.

c. Assume that at each iteration t of AdaBoost, the weak learner returns a hypothesis $h_t$ for which the error $\epsilon_t$ satisfies $\epsilon_t \leq 1/2–\gamma, \gamma > 0$. What is

the probability that the classifier $h_t$ (selected as the best weak learner at iteration t) will be selected again at iteration t+1? Justify your answer.

**Solution.**
a) In the Viola-Jones version of AdaBoost, the distribution $w_{t+1,i}$ is of form:

$$w_{t+1,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \cdot \beta_t^{1-|h_t(x_i)-y_i|} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j} \cdot \beta_t^{|h_t(x_i)-y_i|-1}}.$$

There are two possibilities for every element in the denominator sum:

1. $h_t(x_i) \neq y_i$. The power of $\beta_t$ is 0, so the weight $w_{t,j}$ stays the same.
2. $h_t(x_i) = y_i$. The weight $w_{t,j}$ becomes smaller by dividing it with $\beta_t$.

Note that we used $|h_t(x_i) - y_i|$ to simulate $e_i$.
We will further prove that the final version of the $D_{t+1,i}$ distribution from the Freund-Schapire version of AdaBoost is equal to $w_{t+1,i}$. Before that, however, we need to make a couple of remarks:

- $h_t(x_i) \cdot y_i$ has the same output for $y_i \in \{-1, +1\}$ and $h : \mathbb{R}^n \to \{-1, +1\}$, as $|h_t(x_i) - y_i|$ for $y_i \in \{0, 1\}$ and $h : \mathbb{R}^n \to \{0, 1\}$.
- In both versions, the error $\epsilon_t$ is computed in the same way. Additionally, we will refer to $\frac{\epsilon_t}{1-\epsilon_t}$ as $\beta_t$ for both AdaBoost versions, as it will simplify explanations.

Now we move on to computing $D_{t+1,i}$:

$$\begin{aligned}
D_{t+1,i} &= \frac{D_{t,i}}{Z_{t+1}} \cdot \beta_t^{-\frac{1}{2}h_t(x_i)y_i} \\
&= \frac{D_{t,i}}{\sum_{j=1}^{N} D_{t,j} \cdot \beta_t^{-\frac{1}{2}h_t(x_j)y_j}} \cdot \beta_t^{-\frac{1}{2}h_t(x_i)y_i} \\
&= \frac{D_{t,i}}{\sum_{j=1}^{N} D_{t,j} \cdot \beta_t^{\frac{1}{2}h_t(x_i)y_i - \frac{1}{2}h_t(x_j)y_j}} \\
&= \frac{D_{t,i}}{\sum_{j=1}^{N} D_{t,j} \cdot \beta_t^{\frac{1}{2}(h_t(x_i)y_i - h_t(x_j)y_j)}}.
\end{aligned}$$

In the Freund-Schapire version we have two cases for each element of the denominator sum:

1. $h_t(x_i)y_i = h_t(x_j)y_j$. The power of $\beta_t$ is 0, meaning that the element $D_{t,j}$ stays the same.

2. $h_t(x_i)y_i \neq h_t(x_j)y_j$. Then $\beta_t^{\frac{1}{2}(h_t(x_i)y_i - h_t(x_j)y_j)} = \beta_t^{\frac{1}{2}\cdot(-2)} = \beta_t^{-1}$. Thus the weight $D_{t,j}$ becomes smaller by dividing it with $\beta_t$.

As one can see, case 1 and 2 for the Viola-Jones version map perfectly with their respective cases from the Freund-Schapire version of AdaBoost. Either the weight stays the same, as for case 1, or it is divided by $\beta_t$, as in case 2.

This concludes our proof that $w_{t+1,i}$ is the same as $D_{t+1,i}$.

b) For the Viola-Jones version of AdaBoost, we have:

$$\sum_{t=1}^{T} ln(\beta_t)h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T} ln(\beta_t) \Leftrightarrow$$
$$\sum_{t=1}^{T} ln(\beta_t)h_t(x) \geq \sum_{t=1}^{T} \frac{1}{2}ln(\beta_t) \Leftrightarrow$$
$$\sum_{t=1}^{T} ln(\beta_t)h_t(x) - \sum_{t=1}^{T} \frac{1}{2}ln(\beta_t) \geq 0 \Leftrightarrow$$
$$\sum_{t=1}^{T} ln(\beta_t)h_t(x) - \frac{1}{2}ln(\beta_t) \geq 0 \Leftrightarrow$$
$$\sum_{t=1}^{T} ln(\beta_t) \cdot (h_t(x) - \frac{1}{2}) \geq 0.$$

We can rewrite the Viola-Jones strong classifier as follows:

$$h(x) = \begin{cases} 1, \ \sum_{t=1}^{T} ln(\beta_t) \cdot (h_t(x) - \frac{1}{2}) \geq 0 \\ 0, \ \text{otherwise} \end{cases}$$

We also rewrite the Freund-Schapire strong classifier $h(x) = sign(\sum_{t=1}^{T} w_t h_t(x))$:

$$h(x) = \begin{cases} +1, \ \sum_{t=1}^{T} ln(\beta_t) \cdot \frac{1}{2} \cdot h_t(x) \geq 0 \\ -1, \ \text{otherwise} \end{cases}$$

Since $h_t(x) \in \{0, 1\}$ for the Viola-Jones version and $h_t(x) \in \{-1, +1\}$ for the Freund-Schapire version, this means that $h_t(x) - \frac{1}{2}$ in the Viola-Jones version is equivalent to $\frac{1}{2} \cdot h_t(x)$ in the Freund-Schapire version. The role of both is to determine the sign of $ln(\beta_t), \forall t = \overline{1, T}$ and divide it by 2.

Given that we know from **a)** that the distributions $w_{t,i}$ and $D_{t,i}$ are the same, this means that the values of the betas are the same.

With this, we have proven that the two version of AdaBoost are equivalent.

c)

*Affirmation.* $\mathbb{P}(h_t = h_{t+1}) = 0$.

*Proof.* Assume that $\mathbb{P}(h_t = h_{t+1}) > 0$ and pick the $h_t$ and $h_{t+1}$ such that $h_t = h_{t+1}$, let $m$ be the size of the training set.

We have that

$$\varepsilon_t = \sum_{\substack{1 \leq i \leq m \\ h_t(x_i) \neq y_i}} D_i^{(t)} \tag{5}$$

And, because we do normalization, we have that $D^{(t)}$ is a probability distribution, hence:

$$\begin{aligned} 1 &= \sum_{i=1}^{m} D_i^{(t)} && (6)\\ &= \sum_{\substack{1 \leq i \leq m \\ h_t(x_i) \neq y_i}} D_i^{(t)} + \sum_{\substack{1 \leq i \leq m \\ h_t(x_i) = y_i}} D_i^{(t)} && (7)\\ &= \varepsilon_t + \sum_{\substack{1 \leq i \leq m \\ h_t(x_i) = y_i}} D_i^{(t)} && (8) \end{aligned}$$

Now, consider the error $\varepsilon_{t+1}$:

$$\varepsilon_{t+1} = \sum_{\substack{1 \leq i \leq m \\ h_{t+1}(x_i) \neq y_i}} D_i^{(t+1)} \overset{(17)}{=} \frac{\sum\limits_{\substack{1 \leq i \leq m \\ h_{t+1}(x_i) \neq y_i}} D_i^{(t)}}{\sum\limits_{\substack{1 \leq j \leq m \\ h_t(x_j) = y_j}} D_j^{(t)} \frac{\varepsilon_t}{1-\varepsilon_t} + \sum\limits_{\substack{1 \leq j \leq m \\ h_t(x_j) \neq y_j}} D_j^{(t)}} \tag{9}$$

But we assumed that $h_t = h_{t+1}$, hence:

$$\begin{aligned} \varepsilon_{t+1} &= \frac{\sum\limits_{\substack{1 \leq i \leq m \\ h_t(x_i) \neq y_i}} D_i^{(t)}}{\sum\limits_{\substack{1 \leq j \leq m \\ h_t(x_j) = y_j}} D_j^{(t)} \frac{\varepsilon_t}{1-\varepsilon_t} + \sum\limits_{\substack{1 \leq j \leq m \\ h_t(x_j) \neq y_j}} D_j^{(t)}} && (10)\\ &\overset{(25)}{=} \frac{\varepsilon_t}{\frac{\varepsilon_t}{1-\varepsilon_t} \sum\limits_{\substack{1 \leq j \leq m \\ h_t(x_j) = y_j}} D_j^{(t)} + \varepsilon_t} && (11)\\ &\overset{(28)}{=} \frac{\varepsilon_t}{\frac{\varepsilon_t}{1-\varepsilon_t}(1-\varepsilon_t) + \varepsilon_t} && (12)\\ &= \frac{\varepsilon_t}{2\varepsilon_t} && (13)\\ &= \frac{1}{2} && (14) \end{aligned}$$

But from the hypothesis, we have that $\forall t \in \mathbb{N}$, $\varepsilon_t \leq \frac{1}{2} - \gamma$, where $\gamma > 0$, so $\varepsilon_{t+1} < \frac{1}{2}$, which is a contradiction with (34), so our assumption that $\mathbb{P}(h_t = h_{t+1}) > 0$ is incorrect. It remains, therefore, that $\mathbb{P}(h_t = h_{t+1}) = 0$.

□

**Exercise 2.** Let A be a 24 x 24 matrix representing an image. The integral image of A, denoted by I(A), is the matrix B such that $B_{i,j} = \sum_{i' \leq i, j' \leq j} A_{i,j}$ .

a) Show that I(A) can be calculated from A in linear time in size of A.

b) Show how every Viola and Jones feature can be calculated from I(A) in a constant amount of time (that is, the runtime does not depend on the size of the rectangle defining the feature).

**Solution.**

a) We apply dynamic programming. Let $A \in \mathbb{R}^{\aleph \curvearrowleft \aleph}$. First, observe that filling sequentially each item of the first column and the first row of I(A) can be done in a constant time. Next, given i, j $\geq$ 2, we note that $I(A)_{i,j} = I(A)_{i-1,j} + I(A)_{i,j-1} - I(A)_{i-1,j-1}$. Hence, filling the values of I(A), row by row, can be done in time linear in the size of I(A).

b) We show the calculation of type B. The calculation of the other types is done similarly. Given $i_1 < i_2, j_1 < j_2$, consider the following rectangular regions:

$$A_U = \{(i,j) \mid i_1 \leq i \leq \lfloor \tfrac{i_1+i_2}{2} \rfloor \, j_1 \leq j \leq j_2\}$$

$$A_D = \{(i,j) \mid \lceil \tfrac{i_1+i_2}{2} \rceil \leq i \leq i_2, j_1 \leq j \leq j_2\}$$

Let $b \in \{-1, 1\}$ and h be a hypothesis. That is, $h(A) = b \cdot (\sum_{(i,j) \in A_L} A_{i,j} - \sum_{(i,j) \in A_R} A_{i,j})$. Let $i_3 = \lfloor \frac{i_1+i_2}{2} \rfloor$ and $i_4 = \lceil \frac{i_1+i_2}{2} \rceil$. Then:

$$h(A) = B_{i_2,j_2} - B_{i_3,j_2} - B_{i_2,j_1-1} + B_{i_3,j_1-1} - (B_{i_3,j_2} - B_{i_1-1,j_2} - B_{i_3,j_1-1} + B_{i_1-1,j_1-1})$$

We conclude that h(A) can be calculated in constant time given B = I(A).

**Exercise 3.** Fix $\epsilon \in \left(0, \frac{1}{2}\right)$. Let the training sample be denoted by $m$ points in the plane with $\frac{m}{4}$ negative points all at coordinate $(+1, +1)$, another $\frac{m}{4}$ negative points all at coordinate $(-1, -1)$, $\frac{m(1+\epsilon)}{4}$ positive points all at coordinate $(-1, +1)$, $\frac{m(1-\epsilon)}{4}$ positive points all at coordinate $(+1, -1)$.

a. Describe the behavior of AdaBoost when run on this sample using boosting stumps for the first two rounds.

b. What is the error of the optimal classifier chosen at round 1 in the second round?

**Solution.**

## AdaBoost

- construct distribution $\mathbf{D}^{(t)}$ on $\{1, \ldots, m\}$:
  - $\mathbf{D}^{(1)}(i) = 1/m$
  - given $\mathbf{D}^{(t)}$ and $h_t$: $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}}{Z_{t+1}}$

where $Z_{t+1}$ normalization factor ($\mathbf{D}^{(t+1)}$ is a distribution): $Z_{t+1} = \sum_{i=1}^{m} D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}$

$w_t$ is a weight: $w_t = \dfrac{1}{2} \ln\left(\dfrac{1}{\epsilon_t} - 1\right) > 0$ as the error $\epsilon_t < 0.5$

$\epsilon_t$ is the error of $h_t$ on $\mathbf{D}^{(t)}$ : $\epsilon_t = \Pr_{i \sim D^{(t)}}[h_t(x_i) \neq y_i] = \sum_{i=1}^{m} D^{(t)}(i) \times \mathbb{1}_{[h_t(x_i) \neq y_i]}$

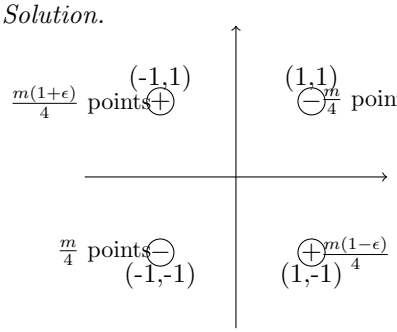
If example $_i$ is correctly classified, then $h_t(_i) =_i$, so at the next iteration $t+1$ its importance (probability distribution) will be decreased to:

$$D^{(t+1)}(i) = \frac{D^{(t)}(i) \times e^{-w_t}}{Z_{t+1}} = \frac{D^{(t)}(i) \times e^{-\frac{1}{2}\ln\left(\frac{1}{\epsilon_t}-1\right)}}{Z_{t+1}} = \frac{D^{(t)}(i) \times \left(\frac{1}{\epsilon_t} - 1\right)^{-\frac{1}{2}}}{Z_{t+1}} = \frac{D^{(t)}(i) \times \sqrt{\frac{\epsilon_t}{1-\epsilon_t}}}{Z_{t+1}}$$

If example $_i$ is misclassified, then $h_t(_i) \neq_i$, so at the next iteration $t+1$ its importance (probability distribution) will be increased to:

$$D^{(t+1)}(i) = \frac{D^{(t)}(i) \times e^{w_t}}{Z_{t+1}} = \frac{D^{(t)}(i) \times e^{\frac{1}{2}\ln\left(\frac{1}{\epsilon_t}-1\right)}}{Z_{t+1}} = \frac{D^{(t)}(i) \times \left(\frac{1}{\epsilon_t} - 1\right)^{\frac{1}{2}}}{Z_{t+1}} = \frac{D^{(t)}(i) \times \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}}{Z_{t+1}}$$

*Solution.*




$$\frac{m(1+\epsilon)}{4} + \frac{m(1-\epsilon)}{4} = \frac{m}{2} \text{ points with } + \text{ label}$$

$$\frac{m}{4} + \frac{m}{4} = \frac{m}{2} \text{ points with } - \text{ label}$$

The probability distribution of the training point $(-1, 1)$ with label $+$ is $\dfrac{\frac{m(1+\epsilon)}{4}}{m} = \dfrac{1+\epsilon}{4}$. For point $(1, -1)$, we obtain $\dfrac{1-\epsilon}{4}$, for points $(1, 1)$ and $(-1, -1)$ with label $-$ we obtain $\frac{1}{4}$.

The initial problem with $m$ points in the training sample is similar with the problem with 4 points with the corresponding probabilities.

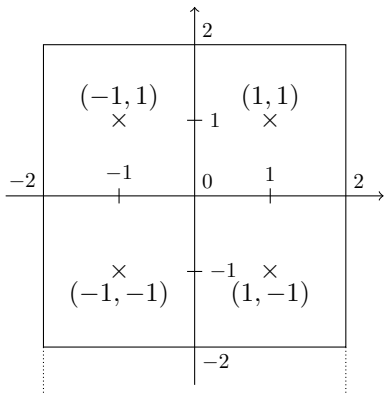
$$S = \left\{ ((-1,+1),\ \underset{\substack{\downarrow \\ point}}{+1}\ ), ((+1,-1),+1), ((+1,+1),-1), ((-1,-1),-1) \right\}$$

$$D^{(1)}\colon \begin{pmatrix} (-1,1) & (1,-1) & (1,1) & (-1,-1) \\ \frac{1+\epsilon}{4} & \frac{1-\epsilon}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

Base hypothesis class = decision stumps in $\mathbb{R}^2$.

$$\mathcal{H}^2_{\mathrm{DS}} = \left\{ h_{i,\theta,b}\colon \mathbb{R}^2 \to \{-1,1\},\ h_{i,\theta,b}(x_1,x_2) = (\theta - x_i)\cdot b \quad \begin{array}{l} 1 \le i \le 2 \\ \theta \in \mathbb{R} \\ b \in \{+1,-1\} \end{array} \right\}$$

= pick a coordinate $i$ (1 or 2), project the input $x = (x_1, x_2)$ on the $i$-th coordinate and obtain $x_i$ if $x_i \le \theta$, label the example $x_i$ with label $b$, else with label $-b$




For our problem, we can see that we can take [a] set of representation thresholds $\theta$ to be $\theta = \{-2, 0, 2\}$. So we have at most 12 base classifiers: $h_{1,-2,1}; h_{1,-2,-1}; \ldots; h_{2,2,-1}$

$+$ | $-$

$h_{1,-2,+1} \to$ project on $x_1$, compare to $-2$, all points $< -2$ get label $+1$, all other get label $-1$

$-$ | $+$

$h_{1,-2,-1} \to$ project on $x_1$, compare to $-2$, all points $< -2$ get label $-1$, all other get label $+1$

$+$ | $-$

$h_{1,+2,+1} \to$ project on $x_1$, compare to $+2$, all points $< +2$ get label $+1$, all other get label $-1$

So we see that on our training set ?? $h_{1,-2,-1}$ and $h_{1,+2,+1}$ will have the same behavior (all points will receive label +1).

If we analyze the behavior of all 12 base classifiers (decision stumps in $\mathbb{R}^2$), we will see that in the end there are only 6 unique base classifiers.

| | | | | | |
|---|---|---|---|---|---|
| $+ \mid +$<br>$+ \mid +$ | $- \mid -$<br>$- \mid -$ | $+ \mid -$<br>$+ \mid -$ | $- \mid +$<br>$- \mid +$ | $- \mid -$<br>$+ \mid +$ | $+ \mid +$<br>$- \mid -$ |
| $h^1$ | $h^2$ | $h^3$ | $h^4$ | $h^5$ | $h^6$ |

So we have $B = \{h^1, h^2, h^3, h^4, h^5, h^6\}$.

Round 1

- distribution $D^{(1)}$: $\begin{pmatrix} (-1,1) & (1,-1) & (1,1) & (-1,-1) \\ \frac{1+\epsilon}{4} & \frac{1-\epsilon}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$

- select the best classifier from $\mathcal{H}$, the one with minimum empirical risk

$$L_{D^{(1)}}(h^1) = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$
$$L_{D^{(1)}}(h^2) = \frac{1+\epsilon}{4} + \frac{1-\epsilon}{4} = \frac{1}{2}$$
$$L_{D^{(1)}}(h^3) = \frac{1}{4} + \frac{1-\epsilon}{4} = \frac{1}{2} - \frac{\epsilon}{4}$$
$$L_{D^{(1)}}(h^4) = \frac{1+\epsilon}{4} + \frac{1}{4} = \frac{1}{2} + \frac{\epsilon}{4}$$
$$L_{D^{(1)}}(h^5) = \frac{1+\epsilon}{4} + \frac{1}{4} = \frac{1}{2} + \frac{\epsilon}{4}$$
$$L_{D^{(1)}}(h^6) = \frac{1}{4} + \frac{1-\epsilon}{4} = \frac{1}{2} - \frac{\epsilon}{4}$$

So, the minimum achievable error is $\frac{1}{2} - \frac{\epsilon}{4}$ and it is attained by base classifiers $h^3$ and $h^6$. Let's choose $h^3$ as our weak classifier: $h^3 = h_{1,0,+1}$.

So, for $t = 1$ (round 1) we have $h_t = h^3 = h_{1,0,-1}$[5].

The error of the base classifier is $\epsilon_1 = \frac{1}{2} - \frac{\epsilon}{4}$.

$$w_1 = \frac{1}{2} \ln\left(\frac{1}{\epsilon_1} - 1\right) = \frac{1}{2}\left(\ln\left(\frac{4}{2-\epsilon} - 1\right)\right) = \ln\left(\frac{2+\epsilon}{2-\epsilon}\right)^{\frac{1}{2}} = \ln\sqrt{\frac{2+\epsilon}{2-\epsilon}}$$

Based on $D^{(1)}$ we will build $D^{(2)}$. Examples correctly classified at round 1 will have now the weight decreased, examples misclassified at round 1 will have their weight increased.

[5] $h_{1,0,+1}$?

$$D^{(2)}((-1,+1)) = \frac{1}{Z_2} D^{(1)}((-1,+1)) \cdot \sqrt{\frac{\epsilon_1}{1-\epsilon_1}} = \frac{1}{Z_2} \cdot \left(\frac{1+\epsilon}{4}\right) \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \quad \searrow$$

$$D^{(2)}((+1,-1)) = \frac{1}{Z_2} \cdot \left(\frac{1-\epsilon}{4}\right) \cdot \sqrt{\frac{2+\epsilon}{2-\epsilon}} \quad \nearrow$$

$$D^{(2)}((+1,+1)) = \frac{1}{Z_2} \cdot \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \quad \searrow$$

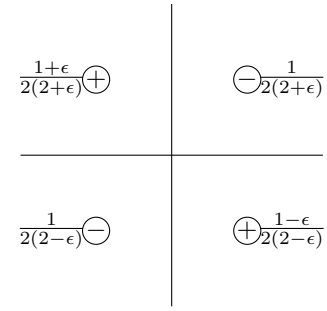
$$D^{(2)}((-1,-1)) = \frac{1}{Z_2} \cdot \frac{1}{4} \cdot \sqrt{\frac{2+\epsilon}{2-\epsilon}} \quad \nearrow$$

We can find the value of $Z_2$ such that $D^{(2)}$ is a probability distribution, meaning that the sum of probability mass should be equal to 1.

$$D^{(2)}((-1,+1)) + D^{(2)}((+1,-1)) + D^{(2)}((+1,+1)) + D^{(2)}((-1,-1)) = 1$$

$$\begin{aligned}
\Rightarrow Z_2 &= \frac{1+\epsilon}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} + \frac{1-\epsilon}{4} \cdot \sqrt{\frac{2+\epsilon}{2-\epsilon}} + \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} + \frac{1}{4} \cdot \sqrt{\frac{2+\epsilon}{2-\epsilon}} \\
&= \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \cdot \left((1+\epsilon) + (1-\epsilon) \cdot \frac{2+\epsilon}{2-\epsilon} + 1 + \frac{2+\epsilon}{2-\epsilon}\right) \\
&= \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \cdot \frac{(1+\epsilon)\cdot(2-\epsilon) + (1-\epsilon)\cdot(2+\epsilon) + (2-\epsilon) + 2 + \epsilon}{2-\epsilon} \\
&= \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \cdot \frac{2+\epsilon-\epsilon^2+2-\epsilon-\epsilon^2+2-\epsilon+2+\epsilon}{2-\epsilon} \\
&= \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \cdot \frac{8-2\epsilon^2}{2-\epsilon} = \frac{1}{4} \cdot \sqrt{\frac{2-\epsilon}{2+\epsilon}} \cdot \frac{2(2-\epsilon)(2+\epsilon)}{2-\epsilon} \\
&= \frac{1}{2} \cdot \sqrt{(2-\epsilon)(2+\epsilon)}
\end{aligned}$$

$$\begin{aligned}
\Rightarrow D^{(2)}((-1,+1)) &= \frac{1+\epsilon}{2(2+\epsilon)} \\
D^{(2)}((+1,-1)) &= \frac{1-\epsilon}{2(2-\epsilon)} \\
D^{(2)}((+1,+1)) &= \frac{1}{2(2+\epsilon)} \\
D^{(2)}((-1,-1)) &= \frac{1}{2(2-\epsilon)}
\end{aligned}$$




What is the error of the base classifier $h^3 = h_{1,0,+1}$ selected at round 1 on $D^{(2)}$?

$$\text{Loss}(h^3) = \frac{1}{2(2-\epsilon)} + \frac{1-\epsilon}{2(2-\epsilon)} = \frac{2-\epsilon}{2(2-\epsilon)} = \frac{1}{2}$$

Round 2

- distribution $D^{(2)}$: $\begin{pmatrix} (-1,1) & (1,-1) & (1,1) & (-1,-1) \\ \frac{1+\epsilon}{2(2+\epsilon)} & \frac{1-\epsilon}{2(2-\epsilon)} & \frac{1}{2(2+\epsilon)} & \frac{1}{2(2-\epsilon)} \end{pmatrix}$

- select the best classifier from $\mathcal{H}$, the one with minimum empirical risk

$$L_{D^{(2)}}(h^1) = \frac{1}{2(2-\epsilon)} + \frac{1}{2(2+\epsilon)} = \frac{2+\epsilon+2-\epsilon}{2(2-\epsilon)(2+\epsilon)} = \frac{2}{(2-\epsilon)(2+\epsilon)} = \frac{4}{2(2-\epsilon)(2+\epsilon)}$$

$$L_{D^{(2)}}(h^2) = \frac{1+\epsilon}{2(2+\epsilon)} + \frac{1-\epsilon}{2(2-\epsilon)} = \frac{(1+\epsilon)\cdot(2-\epsilon)+(1-\epsilon)\cdot(2+\epsilon)}{2(2-\epsilon)(2+\epsilon)} = \frac{4-2\epsilon^2}{2(2-\epsilon)(2+\epsilon)}$$

$$L_{D^{(2)}}(h^3) = \frac{1}{2} = \frac{4-\epsilon^2}{2(2-\epsilon)(2+\epsilon)}$$

$$L_{D^{(2)}}(h^4) = \frac{1}{2} = \frac{4-\epsilon^2}{2(2-\epsilon)(2+\epsilon)}$$

$$L_{D^{(2)}}(h^5) = \frac{1+\epsilon}{2(2+\epsilon)} + \frac{1}{2(2-\epsilon)} = \frac{(1+\epsilon)\cdot(2-\epsilon)+2+\epsilon}{2(2+\epsilon)(2-\epsilon)} = \frac{4+2\epsilon-\epsilon^2}{2(2+\epsilon)(2-\epsilon)}$$

$$L_{D^{(2)}}(h^6) = \frac{1}{2(2+\epsilon)} + \frac{1-\epsilon}{2(2-\epsilon)} = \frac{(2-\epsilon)+(1-\epsilon)\cdot(2+\epsilon)}{2(2+\epsilon)(2-\epsilon)} = \frac{4-2\epsilon-\epsilon^2}{2(2+\epsilon)(2-\epsilon)}$$

The smallest error is attained by $h^6$. This is the base classifier selected at the current round.

So, for $t = 2$ (round 2) we have $h_2 = h^6 = h_{2,0,-1}$.

$$\epsilon_2 = \frac{4-2\epsilon-\epsilon^2}{2(2+\epsilon)(2-\epsilon)}$$

$$w_2 = \frac{1}{2}\ln\left(\frac{1}{\epsilon_2} - 1\right) = \frac{1}{2}\ln\left(\frac{4-\epsilon^2+2\epsilon}{4-\epsilon^2-2\epsilon}\right)$$

□

# 5 Runtime of Learning

**Exercise 1.** Let $\Sigma$ be a finite alphabet and let $\mathcal{X} = \Sigma^m$ be the sample space of all strings of length $m$ over $\Sigma$. Let $\mathcal{H}$ be a hypothesis space over $\mathcal{X}$, where $\mathcal{H} = \{h_w : \Sigma^m \to \{0,1\}, w \in \Sigma^*, 0 < |w| \leq m \text{ s.t. } h_w(x) = 1 \text{ if } w \text{ is a substring of } x\}$. Give an efficient algorithm for finding a hypothesis $h_w$ consistent with a training set in the realizable case. What is the complexity of your algorithm?

**Solution.**

Consider the following algorithm:

**Algorithm 1:** Efficient hypothesis finder

**Result:** The hypothesis $h_w \in \mathcal{H}$ that is consistent with the training set in the realizable case

Let $\Sigma, \mathcal{X}, \mathcal{H}$ be defined as above.
Let $S = \{(w_1, y_1)...(w_n, y_n)\}$ be the training set.
Pick any word $w_k$ from the training set that has label 1.
Assume realizability.
Assume that here is at least one positive sample.
**while** $h_{w^*}$ *not found* **do**
  pick a candidate $h_{w^*}$ where $w^*$ is a substring of $w_k$;
  **if** $w^*$ *is a substring of every* $w_i \in (w_i, 1) \in S$ *that has label* 1 *and is* ***not*** *a substring in any* $w_j \in (w_j, 0) \in S$ *that has label 0* **then**
    return $h_{w^*}$;
  **else**
    continue the loop and pick another $w^*$;
  **end**
**end**

Let $m$ be the length of the words in $S$ (= to the length of the words in $\mathcal{X}$) and $n = |S|$.

To find all the substrings of $w_k$ we need $m^2$ operations - we pick two indices $k1 < k2 \leq len(w_k)$ and we increment them until we get every possible substring $w_k[k1 : k2]$.

Looping through all the words in $S$ requires $n$ operations and for every word we need another $m^2$ operations to do string search in the naïve way - for every $w_i$ we check if the substring $w^*$ exists by doing a sliding window of size 1 until we find the substring - the naïve approach is better explained at [Wik20a], note that there are much better ways to do string search, one can consult [Wik20b] for further references. Be aware that we need to check **all** of the possible words in $S$ such that we make sure that the substring $w^*$ exists in **every** word labeled with 1 and **none** of the ones labeled with 0 - there could be a substring that is common for all the entries in $S$, and picking it would give us false-positives.

We're in the realizable case, so we're guaranteed that there is at least one substring that is common in every word with class 1 and no word with class 0, so the algorithm is ERM.

The final complexity of training is $T_{train} = O(m^2 * n * m^2) = O(m^4 * n)$.

At inference we need $m^2$ operations to label a sample with the hypothesis found during training by doing string search in the naïve way. For a test set of size $k$, we'll have a complexity of $T_{inference} = O(k * m^2) < T_{train}$.

$\mathcal{H}$ has a finite VC-dimension, so according to the Fundamental Theorem of Statistical Learning, we have that $\mathcal{H}$ is PAC-learnable and there are some absolute constants $C_1, C_2$ such that the sample complexity is bounded by:

$$C_1 \frac{d + log(\frac{1}{\delta})}{\varepsilon} \leq m_{\mathcal{H}}(\varepsilon, \delta) \leq C_2 \frac{d\ log(\frac{1}{\varepsilon}) + log(\frac{1}{\delta})}{\varepsilon}$$

Finally, we have that the algorithm is efficient with a (polynomial) runtime that is at least $O(m^4 * C_1 \frac{d+log(\frac{1}{\delta})}{\varepsilon})$ and at most $O(m^4 * C_2 \frac{d\ log(\frac{1}{\varepsilon})+log(\frac{1}{\delta})}{\varepsilon})$.

**Exercise 2.** Let H = $\{[\theta, \infty) \mid \theta \in \mathbb{R}\} \cup \{(-\infty, \theta) \mid \theta \in \mathbb{R}\}$. Give an efficient ERM algorithm for learning H and compute its complexity for each of the following cases:

a) realizable case.

b) agnostic case.

**Solution.**

a) Consider the following algorithm:

```
sort_wrt_coordinates(S)

label_has_changed = False
curr_label = S[0].label

if curr_label == 0:
    i = 1
    while label_has_changed == False and i < len(S):
        if S[i].label != curr_label:
            θ = S[i].x
            label_has_changed = True
        i += 1
    if label_has_changed == False:
        return (−∞, S[0].x)
    return [θ, ∞)
else:
    i = 1
    while label_has_changed == False and i < len(S):
        if S[i].label != curr_label:
            θ = S[i].x
            label_has_changed = True
        i += 1
    if label_has_changed == False:
        return [S[0].x, ∞)
```

```
25        return (−∞, θ)
```

In the algorithm above, S is a tuple vector whose elements are of form (coordinate, label), with len(S) = m.

We proceed by explaining the algorithm step-by-step:

1. Sort S w.r.t. coordinates.
2. Search for the first "label change" among samples. We have two cases here:
   a. The first label is 0, meaning we are searching for a switch from 0 to 1 and will return an interval of form $[\theta, \infty)$.
      If no such "label change" occurs, this means that all labels are 0. We return the interval $(-\infty, S[0].x)$.
   b. The first label is 1, meaning we are searching for a switch from 1 to 0 and will return an interval of form $(-\infty, \theta)$.
      If no such "label change" occurs, this means that all labels are 1. We return the interval $[S[0].x, \infty)$.

The first step takes O(m log m), while the second step is linear in m, so O(m). As a result, our algorithm takes O(m log m) + O(m) time.

We know that VCdim(H) = 2, so according to the *Fundamental Theorem of Statistical Leaning* [SB14], H is PAC-learnable with sample complexity:

$$C_1 \frac{d + log(\delta^{-1})}{\epsilon} \leq m_H(\epsilon, \delta) \leq C_2 \frac{d \cdot log(\epsilon^{-1}) + log(\delta^{-1})}{\epsilon}, C_1, C_2 \in \mathbb{R}.$$

Our runtime is at most $O(C_2 \frac{d \cdot log(\epsilon^{-1}) + log(\delta^{-1})}{\epsilon} log(C_2 \frac{d \cdot log(\epsilon^{-1}) + log(\delta^{-1})}{\epsilon}))$. In other words, the runtime can be written as $O(p(\epsilon^{-1}, \delta^{-1}))$, for some polynomial $p$. This satisfies the definition of an efficient ERM algorithm.

b) Given the data's distribution, our solution for the agnostic case will be searching for an interval with minimal loss. A naïve algorithm would run in $O(m^2)$ and consists of computing the losses for all sample points. However, we can do better with dynamic programming. Consider the following algorithm:

```
1  sort_wrt_coordinates(S)
2
3  open_left = [0] * (len(S) + 1)
4  open_right = [0] * (len(S) + 1)
5  open_left[0] = loss((−∞, S[0].x))
6  open_right[len(S) + 1] = loss([S[len(S)].x + 1, ∞))
7
8  for i in range(1, len(S)):
9      loss_at_i = −1 if S[i].label == 1 else 1
10     open_left[i] = open_left[i − 1] + loss_at_i
```

```
loss_at_i = -1 if S[len(S)].label == 1 else 1
open_left[len(S) + 1] = open_left[len(S)] + loss_at_i

for j in reversed(range(0, len(S))):
    loss_at_j = -1 if S[j].label == 1 else 1
    open_right[j] = open_right[j + 1] + loss_at_j

min_loss_l, min_loss_r = ∞, ∞
left_idx, right_idx = -1, -1

for i in range(0, len(S) + 1):
    if open_left[i] <= min_loss_l:
        min_loss_l = open_left[i]
        left_idx = i
for j in reversed(range(0, len(S) + 1)):
    if open_right[j] <= min_loss_r:
        min_loss_r = open_right[j]
        right_idx = j

if open_left[left_idx] < open_right[right_idx]:
    θ = S[left_idx].x
    return (-∞, θ)
else:
    θ = S[right_idx].x
    return [θ, ∞)
```

In the algorithm above we have:

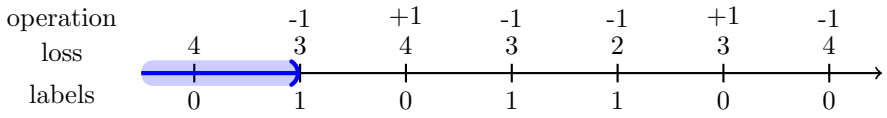
- `S` - tuple vector of type (coordinate, label), with len(S) = m.
- `loss` - $loss(I) = |\{i \in [m] | h_I(x_i) \neq y_i\}|, I \in \{(-\infty, \theta), [\theta, \infty)\}, h_I \in H$.
- `open_left` - list that holds losses for $(-\infty, S[i])$ intervals.
- `open_right` - list that holds losses for $[S[i], \infty)$ intervals.

We proceed by explaining the provided algorithm step-by-step:

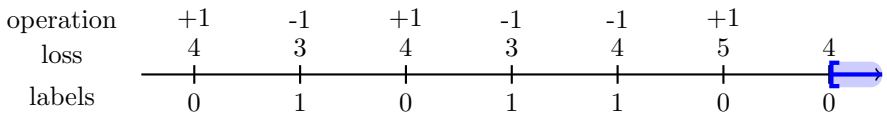
1. Sort S w.r.t. coordinates.
2. Initialize `open_left` and `open_right` as lists of zeroes of length `len(S)+1`. The "`+1`" stands for $(-\infty, S[len(S)] + 1)$ and $[S[len(S)] + 1, \infty)$, respectively.
3. Initialize the first elements of our lists as follows:
    a. `open_left[0] = loss((-∞, S[0].x))`
    b. `open_right[len(open_right) + 1] = loss([S[len(S)].x + 1, ∞))`

**Note:** Computations on `open_right` are done from end to start of S.

4. Compute `open_left[i]` based on `open_left[i-1]`. Add -1 if the $i^{th}$ label is 1 and +1 otherwise.




5. Compute `open_right[j]` based on `open_right[j+1]`. Add -1 if the $j^{th}$ label is 1 and +1 otherwise.




6. Search for the optimal indexes; in other words, those that minimize loss of left-open and right-open intervals. We store the results in `left_idx` and `right_idx`, respectively.

   **Note:** We search for the optimal indexes by using "greater or equal" comparisons. This way, if the loss for $(-\infty, S[i].x)$ is equal to that for $(-\infty, S[i+3].x)$, we will choose the latter as our classifier at that step. Same with $[\theta, \infty)$, but we work from end to start of S.

7. Compare losses at `left_idx` and `right_idx`.

   If `open_left[left_idx]` < `open_right[right_idx]`, return $(-\infty, S[\text{left\_idx}].x)$. Elsewise, return $[S[\text{right\_idx}].x, \infty)$.

Sorting S can be done in O(m log m) and computing the loss for $(-\infty, S[0])$ and $[S[len(S)] + 1, \infty)$ can be done in O(m). Since each addition is done in O(1), filling `open_left` and `open_right` can be done in linear time w.r.t. m. Similarly, searching for the index with minimal loss takes O(m) time.

To conclude, our algorithm takes O(m log m) + O(m) time.

H has finite VCdim, so according to the *Fundamental Theorem of Statistical Leaning* [SB14], H is agnostically PAC-learnable with sample complexity:

$$C_1 \frac{d + log(\delta^{-1})}{\epsilon^2} \leq m_H(\epsilon, \delta) \leq C_2 \frac{d + log(\delta^{-1})}{\epsilon^2}, C_1, C_2 \in \mathbb{R}.$$

Our runtime is at most $O(C_2 \frac{d+log(\delta^{-1})}{\epsilon^2} log(C_2 \frac{d+log(\delta^{-1})}{\epsilon^2}))$. In other words, the runtime can be written as $O(p(\epsilon^{-1}, \delta^{-1}))$, for some polynomial $p$. This satisfies the definition of an efficient ERM algorithm.

# References


[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. USA: Cambridge University Press, 2014. ISBN: 1107057132.

[NY15] Shay Moran Noga Alon and Amir Yehudayoff. "Sign rank versus VC dimension". In: *COLT* 49 (June 2015), pp. 47–80. URL: https://collaborate.princeton.edu/en/publications/sign-rank-versus-vc-dimension.

[Wik20a] Wikipedia contributors. *Naïve string search*. [Online; accessed 20-July-2020]. 2020. URL: https://en.wikipedia.org/wiki/String-searching_algorithm#Na%5C%C3%5C%AFve_string_search.

[Wik20b] Wikipedia contributors. *String-searching algorithm*. [Online; accessed 20-July-2020]. 2020. URL: https://en.wikipedia.org/wiki/String-searching_algorithm.