Lecture 1 – Questions [v1]

Lisa Tse, Antonin Schrab

November 22, 2021

1. **True or False?**

   "For the K-nearest neighbour algorithm, larger K values will tend to lead to overfitting."

   Explain your answer.

   False. Larger k creates more regular, smoother decision boundaries, which could possibly lead to underfitting.

2. **First, give Cover's bound on the Bayes error of the 1-nearest neighbour algorithm as the number of examples goes to infinity. Second, give an example where the 1-nearest neighbour algorithm achieves the Bayes error as the number of examples goes to infinity, explain your reasoning.**

   Two possibilities are when $p^* = 1/C$ (each class is equally likely) or $p^* = 1$ (deterministic case, when the Bayes error=0). In the former case, $\sum_c P(c|x)(1 - P(c|x)) = 1 - \frac{1}{c}$ and $1 - p^* = 1 - \frac{1}{c}$. In the latter case $\sum_c P(c|x)(1 - P(c|x)) = 1 - p^* = 0$.

3. **How can we choose k to ensure that the k-NN makes an unambiguous decision in the two-class case provided the test point is not equidistant to any pair of points in the training set? What values of k give an unambiguous decision under similar assumptions for the 3-class case?**

   2 class case: k is odd (if $k = 2r$ for $r \in \mathbb{N} \setminus \{0\}$ is even then r points labelled from one class and the r other points labelled from the other class).
   3 class case: $k = 1$ (if $k = 2r$ for $r \in \mathbb{N} \setminus \{0\}$ then the same reasoning as in the 2 class case applies, if $k = 2r + 1$ for $r \in \mathbb{N} \setminus \{0\}$ then we can have one point labelled from the third class and the $2r$ remaining points labelled as in the 2 class case).

4. **One of the drawbacks of the nearest-neighbour algorithm is that we must retain all of the training data. Describe a situation where a training point can be removed without affecting the resulting 1-NN classification for any test point in the input space.**

   Start by constructing a Voronoi diagram (see Figure 1), that is, for each point x we construct a region R_x such that every point $y \in R_x$ is closer to x than it is to all the other points.

   Now, if a region R_x of a point x with label ℓ has boundaries only with regions of points with the same label ℓ then we can remove the point x without affecting the resulting 1-NN classification for any test point in the input space. This is because, after having

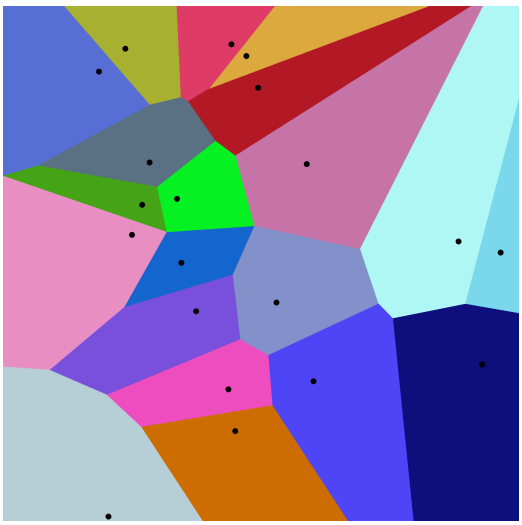


Figure 1: Voronoi diagram

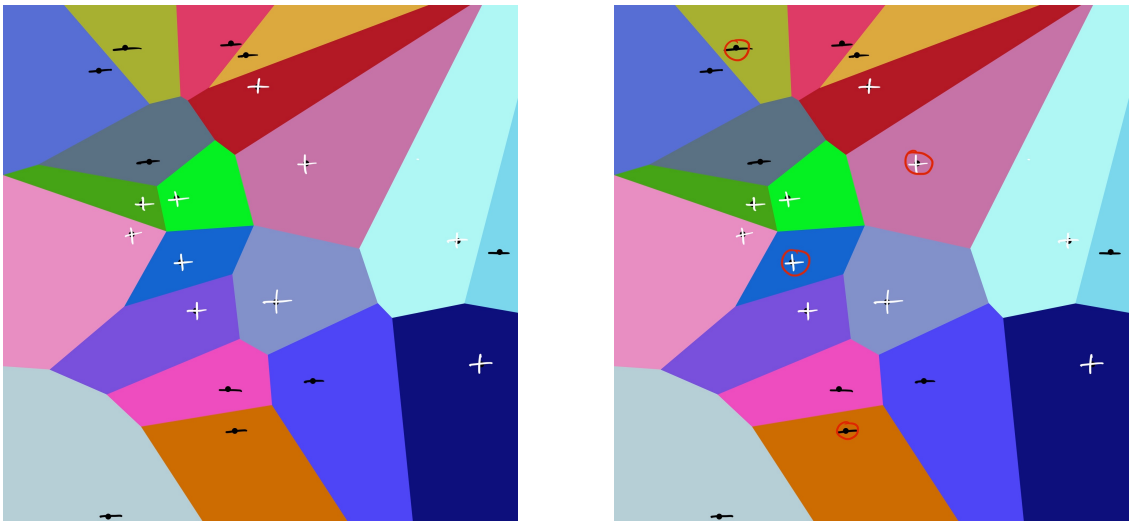


Figure 2: Example

removed x, every point in the region R_x will be closest to one of the points of the regions sharing a boundary with R_x which has the same label.

We provide an example in Figure 2 where we have labelled the points with '+" or '−' (LHS). We are able to remove four points circled in red (RHS) which each belong to a region which has boundaries only with regions of the same label.

5. **In linear regression it is common to transform the training data**

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)$$

to

$$((\boldsymbol{x}_1, 1), y_1), \ldots, ((\boldsymbol{x}_m, 1), y_m)$$

i.e., we add an additional component to each input vector and set it to 1. What is the motivation for this procedure?

To incorporate a bias term.

6. **Given we have a dataset $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$ where $\boldsymbol{x} \in \Re^n$ and $y \in \{1, 2, \ldots, K\}$. Thus the inputs are real vectors and the labels denote one of**

K **classes. How can linear regression (least squares), be used or adapted to tackle such a multi-class classification problem?**

Using one-vs-one (OvO) or one-vs-rest strategies (OvR). For OvR, there are K classifiers and each binary classifier is fed +1 for a class $k \in [K]$, and -1 for all the other classes. It combines the predictions by predicting the class with the highest score $\langle \boldsymbol{w}\boldsymbol{x} \rangle - \theta$. For OvO, each classifier is fed one of the possible combinations of two of the K classes, and predicts according to a majority vote.

7. **Bayes estimator with L^1 loss.** Let $Y \subseteq \mathbb{R}$ and $L(y, \hat{y}) \coloneqq |y - \hat{y}|$. The Bayes estimator is

$$f^* \coloneqq \operatorname*{argmin}_f \mathbb{E}\left[L(y, f(x))\right] = \sum_{x \in X} \sum_{y \in Y} |y - f(x)| p(x, y) = \sum_{x \in X} \left(\sum_{y \in Y} |y - f(x)| p(y|x) \right) p(x)$$

As a function of x, we let $\hat{y} \coloneqq f^*(x)$. Now, set $x = x'$. We then have

$$\sum_{y \in Y} |y - f(x')| p(y|x') p(x') \propto \sum_{y \in Y} |y - f(x')| p(y|x')$$

as $p(x')$ does not depend on y. Since, this is convex, to find the minimum we take the derivative with respect to y and set it equal to 0

$$\begin{aligned}
0 &= \frac{\partial}{\partial \hat{y}} \sum_{y \in Y} |y - \hat{y}| p(y|x') \\
0 &= \frac{\partial}{\partial \hat{y}} \sum_{\substack{y \in Y \\ y \geq \hat{y}}} (y - \hat{y}) p(y|x') + \frac{\partial}{\partial \hat{y}} \sum_{\substack{y \in Y \\ y < \hat{y}}} (\hat{y} - y) p(y|x') \\
0 &= \sum_{\substack{y \in Y \\ y \geq \hat{y}}} -1 \cdot p(y|x') + \sum_{\substack{y \in Y \\ y < \hat{y}}} 1 \cdot p(y|x') \\
0 &= -\mathbb{P}(Y \geq \hat{y} | X = x') + \mathbb{P}(Y < \hat{y} | X = x')
\end{aligned}$$

hence

$$\begin{aligned}
\mathbb{P}(Y \geq \hat{y} | X = x') = \mathbb{P}(Y < \hat{y} | X = x') = 1 - \mathbb{P}(Y \geq \hat{y} | X = x') &\implies \mathbb{P}(Y \geq \hat{y} | X = x') = \frac{1}{2} \\
&\implies \mathbb{P}(Y < \hat{y} | X = x') = \frac{1}{2}.
\end{aligned}$$

By definition, a median of a real random variable Z is a value $m \in \mathbb{R}$ satisfying

$$P(Z \leq m) \geq \frac{1}{2} \qquad \text{and} \qquad P(Z \geq m) \leq \frac{1}{2}.$$

We have $\mathbb{P}(Y \geq \hat{y} | X = x') = \frac{1}{2}$ and $\mathbb{P}(Y \leq \hat{y} | X = x') \geq \mathbb{P}(Y < \hat{y} | X = x') = \frac{1}{2}$, so we conclude that the Bayes estimator $\hat{y}(x)$ is the median of $Y | X = x$.

8. **Adding a bias term: deriving results on page 32.** Recall that $\mathbf{X}$ is a $m \times n$ matrix and that we know, by the result of page 31, that

$$\mathbf{X}^\top \mathbf{X} \mathbf{w} = \mathbf{X}^\top \mathbf{y}.$$

We consider $\widetilde{\mathbf{w}}^\top = (\mathbf{w}^\top, b)$ and $\widetilde{\mathbf{x}}^\top = (\mathbf{x}^\top, 1)$ which gives $\widetilde{\mathbf{X}} = (\mathbf{X}, \mathbf{1})$ for a $m \times 1$ vector $\mathbf{1}$ with all entries equal to 1. Using the result of page 31, we know that

$$\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{X}} \widetilde{\mathbf{w}} = \widetilde{\mathbf{X}}^\top \mathbf{y} \tag{1}$$

where

$$\widetilde{\mathbf{X}}^\top \cong \begin{pmatrix} \mathbf{X}^\top \\ \mathbf{1}^\top \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{X}^\top \mathbf{X} & \mathbf{X}^\top \mathbf{1} \\ \mathbf{1}^\top \mathbf{X} & \mathbf{1}^\top \mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{X}^\top \mathbf{X} & \mathbf{X}^\top \mathbf{1} \\ \mathbf{1}^\top \mathbf{X} & m \end{pmatrix}$$

since $\mathbf{1}^\top \mathbf{1} = \sum_{i=1}^{m} 1 \cdot 1 = m$. We also have

$$\widetilde{\mathbf{X}}^\top \mathbf{y} = \begin{pmatrix} \mathbf{X}^\top \\ \mathbf{1}^\top \end{pmatrix} \mathbf{y} = \begin{pmatrix} \mathbf{X}^\top \mathbf{y} \\ \mathbf{1}^\top \mathbf{y} \end{pmatrix}.$$

Substituting these in Equation (1) we obtain

$$\begin{pmatrix} \mathbf{X}^\top \mathbf{X} & \mathbf{X}^\top \mathbf{1} \\ \mathbf{1}^\top \mathbf{X} & m \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ b \end{pmatrix} = \begin{pmatrix} \mathbf{X}^\top \mathbf{y} \\ \mathbf{1}^\top \mathbf{y} \end{pmatrix}$$

and expanding this matrix multiplication, we get

$$\begin{aligned} \mathbf{X}^\top \mathbf{X} \mathbf{w} + \mathbf{X}^\top b &= \mathbf{X}^\top \mathbf{y}, \\ \mathbf{1}^\top \mathbf{X} \mathbf{w} + mb &= \mathbf{1}^\top \mathbf{y}. \end{aligned}$$

Supervised Learning
Week 2: Kernels and Regularisation

Antonin Schrab

Problem 1.1.

Prove results on page 9.

Problem 1.1.1

If f and g are convex then $f+g$ is convex.

For $p, q \in \mathcal{X}$ and $\alpha \in (0,1)$, we have

$$\begin{aligned}(f+g)(\alpha p + (1-\alpha)q) &= f(\alpha p + (1-\alpha)q) + g(\alpha p + (1-\alpha)q) \\ &\leq \alpha f(p) + (1-\alpha) f(q) + \alpha g(p) + (1-\alpha) g(q) \\ &= \alpha(f(p)+g(p)) + (1-\alpha)(g(p)+g(q)).\end{aligned}$$

This shows that $f+g$ is convex.

Problem 1.1.2

If f is convex and g is affine (linear + a constant) then $f(g(\cdot))$ is convex.

Since g is affine it can be written as $g(x) = a(x+b) = ax + ab$ for some $a, b \in \mathbb{R}$ and for all $x \in \mathbb{R}$. We then have

$$\begin{aligned}(f \circ g)(\alpha p + (1-\alpha)q) &= f(g(\alpha p + (1-\alpha)q)) \\ &= f(\alpha a p + (1-\alpha) a q + ab) \\ &= f(\alpha a(p+b) + (1-\alpha) a(q+b)) \\ &\leq \alpha f(a(p+b)) + (1-\alpha) f(a(q+b)) \\ &= \alpha f(g(p)) + (1-\alpha) f(g(q)).\end{aligned}$$

This shows that $f \circ g$ is convex.

Problem 1.1.3

Suppose M is a symmetric matrix then M is a PSD matrix iff $f(\mathbf{x}) = \mathbf{x}^\top M \mathbf{x}$ is convex.

Suppose M is a symmetric PSD matrix, then it can written as $M = R^\top R$ for some matrix R. We then have

$$f(\mathbf{x}) = \mathbf{x}^\top M \mathbf{x} = \mathbf{x}^\top R^\top R \mathbf{x} = \|R\mathbf{x}\|_2^2$$

and so

$$\begin{aligned}f(\alpha \mathbf{p} + (1-\alpha)\mathbf{q}) &= \|R(\alpha \mathbf{p} + (1-\alpha)\mathbf{q})\|_2^2 \\ &\leq \|\alpha R\mathbf{p}\|_2^2 + \|(1-\alpha)R\mathbf{q})\|_2^2 \\ &= \alpha^2 \mathbf{p}^\top R^\top R \mathbf{p} + (1-\alpha)^2 \mathbf{q}^\top R^\top R \mathbf{q} \\ &= \alpha^2 \mathbf{p}^\top M \mathbf{p} + (1-\alpha)^2 \mathbf{q}^\top M \mathbf{q} \\ &\leq \alpha \mathbf{p}^\top M \mathbf{p} + (1-\alpha) \mathbf{q}^\top M \mathbf{q} \\ &= \alpha f(\mathbf{p}) + (1-\alpha) f(\mathbf{q})\end{aligned}$$

because $\alpha \in (0,1)$ and $(1-\alpha) \in (0,1)$. This shows that f is convex.

Now, suppose f is convex and so it attains its minimum at

$$0 = \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}) = \frac{\partial}{\partial \mathbf{x}} \mathbf{x}^\top M \mathbf{x} = (M + M^\top)\mathbf{x} = 2M\mathbf{x}$$

using the Chain Rule. We deduce that f achieves its minimum at $\mathbf{0}$, so

$$\mathbf{x}^\top M \mathbf{x} = f(\mathbf{x}) \geq f(\mathbf{0}) = \mathbf{0}^\top M \mathbf{0} = 0.$$

This shows that M is positive semidefinite.

Problem 1.1.4

A level set of a convex function is convex.

Let f be a convex function. The level set of f at c is $L \coloneqq \{x : f(x) \leq c\}$. Suppose $p, q \in L$, that is, $f(p) \leq c$ and $f(q) \leq c$. We then have

$$f(\alpha p + (1-\alpha)q) \leq \alpha f(p) + (1-\alpha) f(q) \leq \alpha c + (1-\alpha)c = c,$$

so $\alpha p + (1-\alpha)q \in L$. This proves that the level set L of f is a convex set.

Problem 1.2.

Consider the solution to linear regression optimisation problem. When is it advantageous to compute it via the primal solution? When is it advantageous to compute it via the dual solution? Explain why.

Recall that the matrix X is of size $m \times n$ where we have m samples in $\mathbb{R}^n$. The primal solution (page 16) is

$$\mathbf{w} = (X^\top X + \lambda I_n)^{-1} X^\top y$$

and the dual solution (page 17) is

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i \qquad \text{where} \qquad \boldsymbol{\alpha} = (XX^\top + \lambda I_m)^{-1} y.$$

The operation with the highest time complexity in those formulas are the two matrix inversions. If we have more dimensions n than samples m, it is therefore advantageous to compute via the dual solution. For the case where we have more samples m than dimensions n, it is advantageous to use the primal solution.

Problem 1.3.

Given a kernel $K\colon \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ where $K(\mathbf{x}, \mathbf{t}) \coloneqq (1 + \langle \mathbf{x}, \mathbf{t} \rangle)^2$. Find a feature map $\phi\colon \mathbb{R}^2 \to \mathbb{R}^6$ which corresponds to the kernel.

We have

$$\begin{aligned}
K(\mathbf{x},\mathbf{t}) &= (1+\langle \mathbf{x},\mathbf{t}\rangle)^2 \\
&= (1+x_1t_1+x_2t_2)^2 \\
&= 1+x_1^2t_1^2+x_2^2t_2^2+2x_1t_1+2x_2t_2+2x_1x_2t_1t_2 \\
&= \begin{pmatrix} 1 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \sqrt{2}x_1x_2 \end{pmatrix}^\top \begin{pmatrix} 1 \\ t_1^2 \\ t_2^2 \\ \sqrt{2}t_1 \\ \sqrt{2}t_2 \\ \sqrt{2}t_1t_2 \end{pmatrix} \\
&=: \phi(\mathbf{x})^\top \phi(\mathbf{t}).
\end{aligned}$$

Problem 1.4

For each of the following functions $K\colon \mathbb{R}^2\times\mathbb{R}^2\to\mathbb{R}$ argue whether they are a valid kernel (i.e. the kernel can be written as an inner product in some feature space) and when the answer is positive derive an associated feature map representation.

Problem 1.4.1.

$K(\mathbf{x},\mathbf{t}) = \mathbf{x}^\top D\mathbf{t}$, *where* $D \coloneqq \begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix}$

Note that for $\mathbf{c} = (c_1,c_2)^\top \in \mathbb{R}^2$, we have $\mathbf{c}^\top D\mathbf{c} = c_1^2+5c_2^2 \geq 0$. So, D is positive semidefinite and hence by the result on p39 of the lecture slides, K is a valid kernel. The feature map can be obtained as

$$\begin{aligned}
K(\mathbf{x},\mathbf{t}) &= \mathbf{x}^\top D\mathbf{t} \\
&= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \\
&= x_1t_1+5x_2t_2 \\
&= \begin{pmatrix} x_1 \\ \sqrt{5}x_2 \end{pmatrix}^\top \begin{pmatrix} t_1 \\ \sqrt{5}t_2 \end{pmatrix} \\
&=: \phi(\mathbf{x})^\top \phi(\mathbf{t}).
\end{aligned}$$

Problem 1.4.2

$K(\mathbf{x},\mathbf{t}) = \mathbf{x}^\top D\mathbf{t}$, *where* $D \coloneqq \begin{pmatrix} -1 & 2 \\ 2 & 4 \end{pmatrix}$

Note that, for $\mathbf{x} = (1,0)^\top \in \mathbb{R}^2$ and $c = 1 \in \mathbb{R}$, we have

$$c^2K(\mathbf{x},\mathbf{x}) = \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} -1 & 2 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = -x_1^2+2x_1x_2+2x_1x_2+4x_2^2 = -1 < 0.$$

So the function K is not positive semidefinite, hence, it does not define a valid kernel.

Problem 1.4.3.

$K(\mathbf{x}, \mathbf{t}) = \exp(x_1 t_1)$, *where* x_1 *is the first component of the vector* $\mathbf{x}$ *and, likewise,* t_1 *is the first component of the vector* $\mathbf{t}$.

Using the Taylor series of the exponential function $\exp(u) = \sum_{n=0}^{\infty} \frac{u^n}{n!}$, we obtain

$$\begin{aligned} K(\mathbf{x}, \mathbf{t}) &= \exp(x_1 t_1) \\ &= \sum_{n=0}^{\infty} \frac{x_1^n t_1^n}{n!} \\ &= \begin{pmatrix} 1 \\ x_1 \\ x_2^2/\sqrt{2!} \\ x_3^3/\sqrt{3!} \\ \vdots \end{pmatrix}^\top \begin{pmatrix} 1 \\ t_1 \\ t_2^2/\sqrt{2!} \\ t_3^3/\sqrt{3!} \\ \vdots \end{pmatrix} \\ &=: \phi(\mathbf{x})^\top \phi(\mathbf{t}). \end{aligned}$$

We deduce that K is a valid kernel.

Problem 1.4.4.

$K(\mathbf{x}, \mathbf{t}) = \mathbf{x}^\top \mathbf{t} - (\mathbf{x}^\top \mathbf{t})^2$.

Note that, for $\mathbf{x} = \left(\sqrt{2}, 0\right)^\top \in \mathbb{R}^2$ and $c = 1 \in \mathbb{R}$, we have

$$c^2 K(\mathbf{x}, \mathbf{x}) = 2 - 4 = -2 < 0.$$

So the function K is not positive semidefinite, hence, it does not define a valid kernel.

Problem 1.4.5.

Now prove that if $K : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ *is a given valid kernel then the following transformed kernels are also valid.*

Problem 1.3.5.1

$K(A\mathbf{x}, A\mathbf{t})$, *where* A *is a given* 2×2 *matrix.*

Consider $c_1, \ldots, c_m \in \mathbb{R}$ and $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^2$. Letting $\widetilde{\mathbf{x}}_i \coloneqq A\mathbf{x}_i \in \mathbb{R}^2$ for $i = 1, \ldots, m$. Since K is a kernel, we have

$$\sum_{i=1}^{m} \sum_{j=1}^{m} c_i c_j K(\widetilde{\mathbf{x}}_i, \widetilde{\mathbf{x}}_j)$$

or equivalently

$$\sum_{i=1}^{m} \sum_{j=1}^{m} c_i c_j K(A\mathbf{x}_i, A\mathbf{x}_j).$$

This proves that $K(A\mathbf{x}, A\mathbf{t})$ is positive semidefinite and hence it is a valid kernel.

Problem 1.4.5.2.

$f(\mathbf{x})K(\mathbf{x},\mathbf{t})f(\mathbf{t})$, where f is a given real-valued function.

Consider $c_1,\dots,c_m \in \mathbb{R}$ and $\mathbf{x}_1,\dots,\mathbf{x}_m \in \mathbb{R}^2$. Let $\widetilde{c}_i \coloneqq f(\mathbf{x}_i)c_i \in \mathbb{R}$ for $i = 1,\dots,m$. Since K is a kernel, we have

$$\sum_{i=1}^{m}\sum_{j=1}^{m} \widetilde{c}_i\widetilde{c}_j K(\mathbf{x}_i,\mathbf{x}_j)$$

or equivalently

$$\sum_{i=1}^{m}\sum_{j=1}^{m} c_i c_j f(\mathbf{x}_i)K(A\mathbf{x}_i,A\mathbf{x}_j)f(\mathbf{x}_j).$$

This proves that $f(\mathbf{x})K(\mathbf{x},\mathbf{t})f(\mathbf{t})$ is positive semidefinite and hence it is a valid kernel.

Problem 1.5.

Consider a Gaussian kernel function $K\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined by $K(\mathbf{x},\mathbf{z}) \coloneqq e^{-\|\mathbf{x}-\mathbf{z}\|^2}$, does there exist a finite-dimensional feature map representation? I.e., does there exist a $\phi\colon \mathbb{R}^n \to \mathbb{R}^d$ such that $K(\mathbf{x},\mathbf{z}) = \langle\phi(\mathbf{x}),\phi(\mathbf{z})\rangle$? Indicate an answer "yes" or "no" and provide an argument supporting your answer. [hard]

No.

Suppose for contradiction that there exist a finite-dimensional feature map representation, that is there exists $\phi\colon \mathbb{R}^n \to \mathbb{R}^d$ for some finite $d \in \mathbb{N}$ such that

$$e^{-\|\mathbf{x}-\mathbf{z}\|^2} = K(\mathbf{x},\mathbf{z}) = \sum_{i=1}^{d} \phi_i(\mathbf{x})\phi_i(\mathbf{z})$$

Consider the points

$$\mathbf{x}_\ell \coloneqq (\ell, 0, \dots, 0) \in \mathbb{R}^n$$

for $\ell \in \mathbb{N}\setminus\{0\}$. For $\ell \neq h$, the angle θ between $\phi(\mathbf{x}_\ell)$ and $\phi(\mathbf{x}_h)$ in $\mathbb{R}^d$ satisfies

$$\cos\theta = \frac{\phi(\mathbf{x}_\ell)^\top\phi(\mathbf{x}_h)}{(\phi(\mathbf{x}_\ell)^\top\phi(\mathbf{x}_\ell))(\phi(\mathbf{x}_h)^\top\phi(\mathbf{x}_h))} = \frac{K(\mathbf{x}_\ell,\mathbf{x}_h)}{K(\mathbf{x}_\ell,\mathbf{x}_\ell)K(\mathbf{x}_h,\mathbf{x}_h)} = e^{-|\ell-h|} \leq e^{-1}$$

Since arccos is a decreasing function on $[-1,1]$, we obtain that the angle satisfies

$$\theta \geq \arccos(e^{-1}) > 0.$$

This means that we have infinitely many vectors $\{\phi(\mathbf{x}_\ell) : \ell \in \mathbb{N}\setminus\{0\}\}$ in $\mathbb{R}^d$ with the angle between any pair of these being greater than $e^{-1} > 0$. This is a contradiction as there exist only finitely many such vectors (see next two paragraphs). Having reached a contradiction, we deduce that our hypothesis that there exists a finite-dimensional feature map was wrong. We have proved that there does not exist such a finite-dimensional feature map.

Consider a set of vectors $\mathcal{S}_\epsilon$ in $\mathbb{R}^d$, satisfying the property that the angle between any two vectors in $\mathcal{S}_\epsilon$ is greater than some positive constant $\epsilon > 0$. The length of the vectors does not influence the angles between them, hence we can assume that they are of unit lengths, so the vectors lie on the d-dimensional unit sphere. The fact that $\mathcal{S}_\epsilon$ must be finite is intuitive (at least for $d = 2$ and $d = 3$) and can be assumed, we prove it for completeness.

Consider the d-dimensional unit sphere. Since it is a contained within $[-1,1]^d$, the volume of this sphere is bounded by 2^d (i.e. the volume of $[-1,1]^d$). Now consider d vectors $\mathbf{v}_1,\dots,\mathbf{v}_d \in \mathcal{S}_\epsilon$

lying on the unit sphere. Since the angles between any two such vectors is greater than $\epsilon > 0$, the polytope with vertices $\{\mathbf{v}_1, \dots, \mathbf{v}_d, \mathbf{0}\}$ in $\mathbb{R}^d$ must have volume greater than some positive value V_ϵ depending on ϵ. So, d vectors in $\mathcal{S}_\epsilon$ contribute at least volume V_ϵ. Using a crude upper bound, $d\left(\left\lceil \frac{2^d}{V_\epsilon} \right\rceil + 1\right)$ vectors in $\mathcal{S}_\epsilon$ form at least $\left(\left\lceil \frac{2^d}{V_\epsilon} \right\rceil + 1\right)$ non-overlapping polytopes constructed as above. Those contribute at least volume $\left(\left\lceil \frac{2^d}{V_\epsilon} \right\rceil + 1\right) V_\epsilon$ which is larger than the upper bound 2^d on the volume of the unit sphere. This is a contradiction since by adding volumes of non-overlapping polytopes contained within the unit sphere, we obtain larger volume than the the volume of the sphere itself. We deduce that $\mathcal{S}_\epsilon \leq d\left(\left\lceil \frac{2^d}{V_\epsilon} \right\rceil + 1\right) < \infty$.

Problem 2.1.

Argue that the vector space definition implies that every element $\mathbf{x} \in X$ has an additive inverse $-\mathbf{x} \in X$ such that $\mathbf{x} + (-\mathbf{x}) = \mathbf{0}$.

We have

$$\mathbf{0} = 0\mathbf{x} = (1-1)\mathbf{x} = 1\mathbf{x} + (-1)\mathbf{x} = \mathbf{x} + (-1)\mathbf{x}$$

where we used property 7 for the first equality, properties of $\mathbb{R}$ for the second equality, property 5 for the third equality and property 7 for the last equality (properties of vector spaces defined on page 3). We deduce that the additive inverse of $\mathbf{x} \in X$ is $(-1)\mathbf{x} \in X$ which we simply write as $-\mathbf{x} \in X$.

Problem 2.2. Kernels between sets.

Let X be a finite set. Define $K \colon 2^X \times 2^X \to \mathbb{R}$ as

$$K(A, B) \coloneqq 2^{|A \cap B|}$$

where $A, B \subseteq X$. Prove that K is a kernel.

Recall that 2^X is the power set of X (i.e. the set of all subsets of X, including the empty set), and that $|2^X| = 2^{|X|}$.

To prove that K is a kernel, we construct a feature map. For a statement S which is either true or false, we use the notation

$$[S] = \begin{cases} 1 & \text{if } S \text{ is true,} \\ 0 & \text{if } S \text{ is false.} \end{cases}$$

Now, define the features

$$\phi_C(A) = [A \subseteq C]$$

indexed by subsets $C \in 2^X$ (as opposed to the more conventional indices $i = 1, 2, 3, \dots$). For $A, B \in 2^X$, we then have

$$\begin{aligned} \sum_{C \in 2^X} \phi_C(A)\phi_C(B) &= \sum_{C \in 2^X} [C \subseteq A][C \subseteq B] = \sum_{C \in 2^X} [C \subseteq A \cap B] \\ &= \sum_{C \in 2^{A \cap B}} 1 + \sum_{C \in 2^X \setminus 2^{A \cap B}} 0 = \left|2^{A \cap B}\right| = 2^{|A \cap B|} = K(A, B). \end{aligned}$$

This proves that K is indeed a kernel.

Problem 2.3. Min kernel.

Problem 2.3.1.

Argue that $\min(x,t)$ *(where* $x,t \in [0,\infty)$*) is a kernel for "a more complicated example" on page 7. See discussion on page 36, on going from a kernel to a Hilbert space. [technical]*

Consider the inner product space $\mathcal{F}$ which consists of functions from $[0,\infty)$ to $\mathbb{R}$ satisfying

1. $f(0) = 0$
2. f is absolutely continuous (hence $f(b) - f(a) = \int_a^b f'(u)\mathrm{d}u$)
3. $\int_0^\infty (f'(u))^2 \mathrm{d}u < \infty$

with the inner product

$$\langle f, g\rangle := \int_0^\infty f'(u)g'(u)\mathrm{d}u$$

for $f, g \in \mathcal{F}$.

We consider the feature map $\phi : [0,\infty) \to \mathcal{F}$ defined as $\phi(x) \coloneqq \Phi_x \in \mathcal{F}$ for all $x \geq 0$, where

$$\Phi_x(u) := \begin{cases} u & \text{if } u \leq x, \\ x & \text{if } u > x, \end{cases}$$

for all $u \geq 0$. It can easily be verified that $\Phi_x \in \mathcal{F}$ for all $x \geq 0$ (i.e. that it satisfies the three conditions). Note that, when we differentiate Φ_x for some $x \geq 0$, we obtain the step-function

$$\Phi_x'(u) = \begin{cases} 1 & \text{if } u \leq x, \\ 0 & \text{if } u > x, \end{cases}$$

for all $u \geq 0$. Note that, for $x, t \in \mathbb{R}$, we have

$$\Phi_x'(u)\Phi_t'(u) = \begin{cases} 1 & \text{if } u \leq \min(x,t), \\ 0 & \text{if } u > \min(x,t), \end{cases}$$

for all $u \geq 0$. Finally, for $x, t \in \mathbb{R}$, we have

$$\langle \phi(x), \phi(t)\rangle = \langle \Phi_x, \Phi_t\rangle = \int_0^\infty \Phi_x'(u)\Phi_t'(u)\mathrm{d}u = \int_0^{\min(x,t)} 1\,\mathrm{d}u + \int_{\min(x,t)}^\infty 0\,\mathrm{d}u = \min(x,t)$$

This proves that $\min(x,t)$ is a kernel.

Problem 2.3.2. Determining an explicit feature map.

Let $M > 0$*. Find a set of basis functions* $\phi_\ell : [0,M] \to \mathbb{R}$ *for* $\ell \in \mathbb{N}$ *such that*

$$\min(x,t) = \sum_{\ell=0}^\infty \phi_\ell(x)\phi_\ell(t).$$

[technical, very difficult]

We work with the kernel $k(x,t) \coloneqq \min(x,t)$ for $x, t \in [0,M]$. Consider the integral operator $T_k : L^2[0,M] \to L^2[0,M]$ which for $f \in L^2[0,M]$, is defined as

$$(T_k f)(x) \coloneqq \int_0^M k(x,u)f(u)\mathrm{d}u.$$

Mercer's Theorem states that, $x, t \in [0, M]$, we have

$$k(x,t) = \sum_{\ell=0}^{\infty} \lambda_\ell e_\ell(x) e_\ell(t)$$

where $\{e_\ell\}_{\ell=0}^{\infty}$ is an *orthonormal basis* for $L^2[0,M]$ consisting of eigenfunctions of the operator T_k with associated eigenvalues $\{\lambda_\ell\}_{\ell=0}^{\infty}$. Note that, letting

$$\phi_\ell \coloneqq \sqrt{\lambda_\ell} e_\ell \in L^2[0,M]$$

for $\ell \in \mathbb{N}$, we then have

$$\sum_{\ell=0}^{\infty} \phi_\ell(x)\phi_\ell(t) = \sum_{\ell=0}^{\infty} \sqrt{\lambda_i} e_\ell(x) \sqrt{\lambda_\ell} e_\ell(t) = \sum_{\ell=0}^{\infty} \lambda_\ell e_\ell(x) e_\ell(t) = k(x,t)$$

for all $x, t \in [0, M]$. Hence, if we obtain the eigenvalues and eigenfunctions of the operator T_k, we can recover the feature map of the kernel k.

We now explicitly derive the eigenpairs for the kernel $k(x,t) \coloneqq \min(x,t)$ for $x, t \in [0,M]$. An eigenpair (λ, e) is by definition a scalar $\lambda \in \mathbb{R}$ and a function $e \in L^2[0,M]$ satisfying for all $x \in [0,M]$

$$\begin{aligned}
\lambda e(x) &= (T_k e)(x) \\
&= \int_0^M \min(x,t) e(t) \mathrm{d}t \\
&= \int_0^x \min(x,t) e(t) \mathrm{d}t + \int_x^M \min(x,t) e(t) \mathrm{d}t \\
&= \int_0^x t e(t) \mathrm{d}t + \int_x^M x e(t) \mathrm{d}t
\end{aligned}$$

Letting $x = 0$, we get $\lambda e(0) = 0$ and deduce that $e(0) = 0$. We now differentiate both sides with respect to x using Leibniz integral rule and the Chain rule, we obtain

$$\begin{aligned}
\lambda e'(x) &= \left(\frac{\partial}{\partial x} \int_0^x t e(t) \mathrm{d}t\right) + x \frac{\partial}{\partial x}\left(\int_x^M e(t) \mathrm{d}t\right) + \left(\frac{\partial x}{\partial x}\right)\left(\int_x^M e(t) \mathrm{d}t\right) \\
&= x e(x) + x(-e(x)) + 1 \int_x^M e(t) \mathrm{d}t \\
&= \int_x^M e(t) \mathrm{d}t
\end{aligned}$$

Letting $x = M$, we get $\lambda e'(M) = 0$ and deduce that $e'(M) = 0$. Differentiating with respect to x and using Leibniz integral rule one more time, we get

$$\lambda e''(x) = -e(x).$$

This is a second order differential equation which has solution

$$e(x) = c_1 \cos\left(\frac{x}{\sqrt{\lambda}}\right) + c_2 \sin\left(\frac{x}{\sqrt{\lambda}}\right)$$

for some constants $c_1, c_2 \in \mathbb{R}$. Note that $c_1 = e(0) = 0$ and

$$0 = e'(M) = \frac{c_2}{\sqrt{\lambda}} \cos\left(\frac{M}{\sqrt{\lambda}}\right)$$

which has solutions

$$\frac{M}{\sqrt{\lambda_\ell}} = \left(\ell + \frac{1}{2}\right)\pi \qquad \Longleftrightarrow \qquad \lambda_\ell = \frac{M^2}{\left(\ell + \frac{1}{2}\right)^2 \pi^2}$$

for $\ell \in \mathbb{N}$. We have

$$e_\ell(x) = c_2 \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} x\right)$$

for $\ell \in \mathbb{N}$, since we want an orthonormal basis, we require

$$1 = \|e_\ell\|_2^2 = \int_0^M e_\ell(x)^2 \mathrm{d}x = c_2^2 \int_0^M \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} x\right)^2 \mathrm{d}x = c_2^2 \frac{M}{2} \qquad \Longleftrightarrow \qquad c_2 = \sqrt{\frac{2}{M}}.$$

For $\ell \in \mathbb{N}$, the eigenpairs are then

$$\lambda_\ell = \frac{M^2}{\left(\ell + \frac{1}{2}\right)^2 \pi^2} \qquad \text{and} \qquad e_\ell(x) = \sqrt{\frac{2}{M}} \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} x\right)$$

which give the feature map

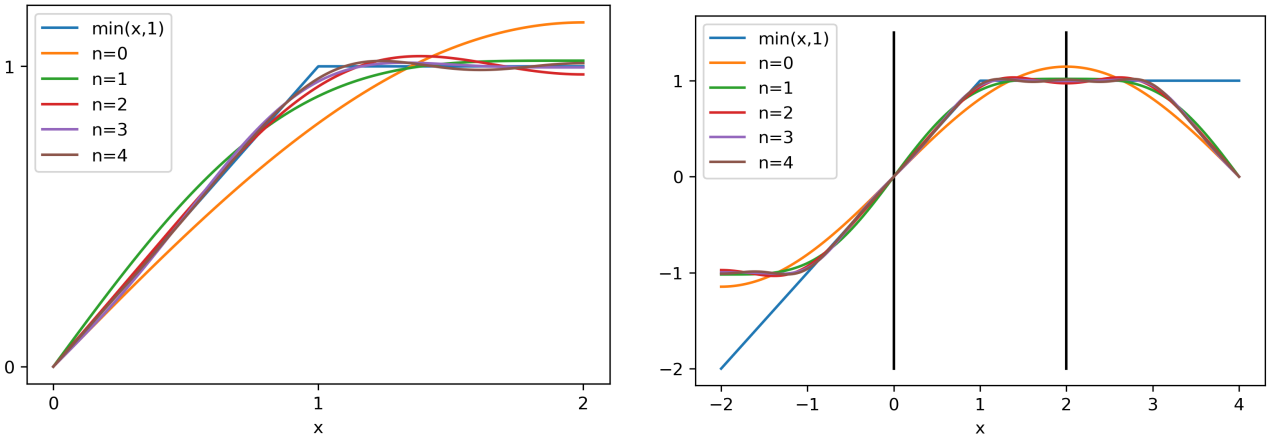
$$\phi_\ell(x) \coloneqq \sqrt{\lambda_\ell} e_\ell(x) = \frac{\sqrt{2M}}{\left(\ell + \frac{1}{2}\right)\pi} \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} x\right).$$

for $x \in [0, M]$ We have proved that

$$k(x,t) = \sum_{\ell=0}^{\infty} \phi_\ell(x)\phi_\ell(t) = \sum_{\ell=0}^{\infty} \frac{2M}{\left(\ell + \frac{1}{2}\right)^2 \pi^2} \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} x\right) \sin\left(\frac{\left(\ell + \frac{1}{2}\right)\pi}{M} t\right)$$

for all $x, t \in [0, M]$.

To illustrate that this is the correct answer, we can plot the partial sum (replacing $\sum_{\ell=0}^{\infty}$ with $\sum_{\ell=0}^{n}$) for $n = 0, 1, 2, 3, 4$ with $M = 2$ and $t = 1$.



SVM Problems

November 25, 2021

1 Problems 1

1. Choose the separating hyperplane (i.e. an affine plane which has all positive examples on one side and all negative examples on the other) that maximises the distance of the nearest example to it.

2. If the removed example is a support vector (2.1) then the margin will (normally) increase (and never decrease). Otherwise (2.2) there will be no change in the optimal hyperplane and its margin.

3. The term $\|\boldsymbol{w}\|^2$ is the regulariser and $1/C$ is the regularisation coefficient. The term $\sum_{i\in[m]} \xi_i$ is the loss function.

2 Problems 2

1. The function $\frac{1}{2}\boldsymbol{w}^\intercal\boldsymbol{w}$ is strictly convex. Also, for all $i \in [m]$ we have that the set $\{\boldsymbol{w} \mid y_i\boldsymbol{w}^\intercal\boldsymbol{x}_i \geq 1\}$ is a half space which is convex. Since the intersection of a collection of convex sets is itself convex we then have that the set $\{\boldsymbol{w} \mid \forall i \in [m]\,,\, y_i\boldsymbol{w}^\intercal\boldsymbol{x}_i \geq 1\}$ is convex, and from the problem description is non-empty. Our problem is then to minimise a strictly convex function over a non-empty convex set which always has an unique solution.

The geometric meaning of the solution is that it is the separating hyperplane, going through the origin, of which the distance of the nearest example to it is maximised.

2. Using the method of Lagrange multipliers we need to find a saddle point of:

$$L = \frac{1}{2}\boldsymbol{w}^\intercal\boldsymbol{w} - \sum_{i\in[m]} c_i[y_i\boldsymbol{w}^\intercal\boldsymbol{x}_i - 1]$$

minimising L over $\boldsymbol{w}$ and maximising over $\boldsymbol{c}$ with $\boldsymbol{c} \geq 0$. We have:

$$\nabla_{\boldsymbol{w}} L = \boldsymbol{w} - \sum_{i\in[m]} c_i y_i \boldsymbol{x}_i$$

so to minimise L over $\boldsymbol{w}$ we set $\nabla_{\boldsymbol{w}} L = 0$ which gives us:

$$\boldsymbol{w} = \sum_{i \in [m]} c_i y_i \boldsymbol{x}_i$$

3. Continuing on from question 2 we now need to maximise L over $\boldsymbol{c}$ with $\boldsymbol{c} \geq \boldsymbol{0}$. From question 2 we obtained:

$$\boldsymbol{w} = \sum_{j \in [m]} c_j y_j \boldsymbol{x}_j = \sum_{k \in [m]} c_k y_k \boldsymbol{x}_k$$

so substituting back into the equation for L we have:

$$\begin{aligned}
L &= \frac{1}{2} \boldsymbol{w}^\intercal \boldsymbol{w} - \sum_{i \in [m]} c_i [y_i \boldsymbol{w}^\intercal \boldsymbol{x}_i - 1] \\
&= \frac{1}{2} \boldsymbol{w}^\intercal \boldsymbol{w} - \sum_{i \in [m]} c_i y_i \boldsymbol{w}^\intercal \boldsymbol{x}_i + \sum_{i \in [m]} c_i \\
&= \frac{1}{2} \left(\sum_{j \in [m]} c_j y_j \boldsymbol{x}_j \right)^\intercal \left(\sum_{k \in [m]} c_k y_k \boldsymbol{x}_k \right) - \sum_{i \in [m]} c_i y_i \left(\sum_{j \in [m]} c_j y_j \boldsymbol{x}_j \right)^\intercal \boldsymbol{x}_i + \sum_{i \in [m]} c_i \\
&= \frac{1}{2} \left(\sum_{j \in [m]} c_j y_j \boldsymbol{x}_j^\intercal \right) \left(\sum_{k \in [m]} c_k y_k \boldsymbol{x}_k \right) - \sum_{i \in [m]} c_i y_i \left(\sum_{j \in [m]} c_j y_j \boldsymbol{x}_j^\intercal \right) \boldsymbol{x}_i + \sum_{i \in [m]} c_i \\
&= \frac{1}{2} \sum_{j \in [m]} \sum_{k \in [m]} c_j y_j \boldsymbol{x}_j^\intercal c_k y_k \boldsymbol{x}_k - \sum_{i \in [m]} \sum_{j \in [m]} c_i y_i c_j y_j \boldsymbol{x}_j^\intercal \boldsymbol{x}_i + \sum_{i \in [m]} c_i \\
&= \frac{1}{2} \sum_{i \in [m]} \sum_{j \in [m]} c_i y_i c_j y_j \boldsymbol{x}_j^\intercal \boldsymbol{x}_i - \sum_{i \in [m]} \sum_{j \in [m]} c_i y_i c_j y_j \boldsymbol{x}_j^\intercal \boldsymbol{x}_i + \sum_{i \in [m]} c_i \\
&= -\frac{1}{2} \sum_{i \in [m]} \sum_{j \in [m]} c_i y_i c_j y_j \boldsymbol{x}_j^\intercal \boldsymbol{x}_i + \sum_{i \in [m]} c_i
\end{aligned}$$

which must be maximised over all $\boldsymbol{c} \geq 0$. This gives us the first part of the solution (P2). The derivative of this, with respect to c_i, is:

$$- \sum_{j \in [m]} c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j + 1$$

so at the minimiser of (P2) we have, for all $i \in [m]$, that $c_i = 0$ or

$$\sum_{j \in [m]} c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j = 1$$

Now, from Q2 we have that:

$$\boldsymbol{w} = \sum_{i \in [m]} c_i y_i \boldsymbol{x}_i$$

so

$$\boldsymbol{w}^\intercal \boldsymbol{w} = \sum_{i\in[m]} \sum_{j\in[m]} c_i c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j = \sum_{i\in[m]} c_i \sum_{j\in[m]} c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j$$

we now consider the term $c_i \sum_{j\in[m]} c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j$. If $c_i = 0$ then this term is equal to 0 so equal to c_i. Otherwise we have, from above, that $\sum_{j\in[m]} c_j y_i y_j \boldsymbol{x}_i^\intercal \boldsymbol{x}_j = 1$ so the term is again equal to c_i . The result is then obtained by summing over $i \in [m]$.

Lecture 4 – Questions

December 3, 2021

1 Problems – 1

1. *Given the dataset*

$$\{((1,1),9),((1,2),-4),((1,3),2),((2,2),4),((2,3),2)\}$$

find the regression tree corresponding to the greedy recursive partitioning procedure with respect to square error.

Let's stop the procedure when there are at most 2 points in each partition. **First split**

- $x_1 \leq 1$, $x_1 > 1$:

  $R_1 = \{\boldsymbol{x} : x_1 \leq 1\}$: we have the points $\{((1,1),9),((1,2),-4),((1,3),2)\}$
  $c_1 = \frac{9-4+2}{3} = \frac{7}{3}$

  $R_2 = \{\boldsymbol{x} : x_1 > 1\}$: we have the points $((2,2),4),((2,3),2)$
  $c_2 = \frac{4+2}{2} = 3$

$$Error = \left(9 - \frac{7}{3}\right)^2 + \left(-4 - \frac{7}{3}\right)^2 + \left(2 - \frac{7}{3}\right)^2 + (4-3)^2 + (2-3)^2 \approx 86.7$$

- $x_2 \leq 1$, $x_2 > 1$:

  $R_1 = \{\boldsymbol{x} : x_2 \leq 1\}$: we have the points $((1,1),9)$
  $c_1 = 9$

  $R_2 = \{\boldsymbol{x} : x_2 > 1\}$: we have the points $((1,2),-4),((1,3),2),((2,2),4),((2,3),2)$
  $c_2 = \frac{-4+2+4+2}{4} = 1$

$$Error = (9-9)^2 + (-4-1)^2 + (2-1)^2 + (4-1)^2 + (2-1)^2 = 36$$

- $x_2 \leq 2$, $x_2 > 2$:

  $R_1 = \{\boldsymbol{x} : x_2 \leq 2\}$: we have the points $((1,1),9),((1,2),-4),((2,2),4)$
  $c_1 = \frac{9-4+4}{3} = 3$

  $R_2 = \{\boldsymbol{x} : x_2 > 1\}$: we have the points $((1,3),2),((2,3),2)$
  $c_2 = 2$

$$Error = (9-3)^2 + (-4-3)^2 + (4-3)^2 + (2-2)^2 + (2-2)^2 = 86$$

We pick the partition $x_2 \leq 1$, $x_2 > 1$ for our first split.

Second split

- $x_1 \leq 1$, $x_1 > 1$:

  $R_1 = \{\boldsymbol{x} : (x_2 \leq 1)\}$: we have the points $\{((1,1),9)\}$
  $c_1 = 9$

  $R_2 = \{\boldsymbol{x} : (x_2 > 1) \wedge (x_1 \leq 1)\}$: we have the points $\{((1,2),-4),((1,3),2)\}$
  $c_2 = \frac{-4+2}{2} = -1$

  $R_3 = \{\boldsymbol{x} : (x_2 > 1) \wedge (x_1 > 1)\}$: we have the points $((2,2),4),((2,3),2)$
  $c_2 = \frac{4+2}{2} = 3$

  $$Error = (9-9)^2 + (-4+1)^2 + (2+1)^2 + (4-3)^2 + (2-3)^2 = 20$$

- $x_2 \leq 2$, $x_2 > 2$:

  $R_1 = \{\boldsymbol{x} : (x_2 \leq 1)\}$: we have the points $\{((1,1),9)\}$
  $c_1 = 9$

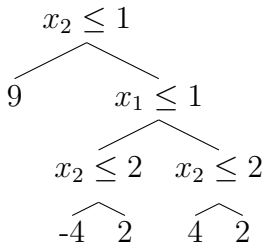
  $R_2 = \{\boldsymbol{x} : (x_2 > 1) \wedge (x_2 \leq 2)\}$: we have the points $\{((1,2),-4),((2,2),4)\}$
  $c_2 = \frac{-4+4}{2} = 0$

  $R_3 = \{\boldsymbol{x} : x_2 > 2\}$: we have the points $((1,3),2),((2,3),2)$
  $c_2 = 2$

  $$Error = (9-9)^2 + (-4-0)^2 + (4-0)^2 + (2-2)^2 + (2-2)^2 = 32$$

We pick the partition $x_1 \leq 1$, $x_1 > 1$ for our second split.

The resulting tree looks like



2. *Explain how a random forest is constructed. What is the motivation for the tree construction method?*

   The 'wisdom of crowds' argument for ensemble learning is strongest when each of the members of the crowds makes independent predictions. The random forest method is designed to encourage each decision tree component of the random forest to be uncorrelated.

   - Draw M samples from the training data (sample with replacement)
   - Consider a subset of the features from this sample ($n_{tree} < n$ where n is the total number of features - typically $n_{tree} = \sqrt{n}$ or $\log(n)$)
   - Build a decision tree
   - Repeat 1-2-3 N times to generate N trees

- For regression our prediction is the mean of the outputs of each tree in the random forest, for classification our prediction is the majority vote.

Both the (1) bootstrap sampling and (2) random feature selection contribute to decorrelating each tree.

Decision trees are typically high variance, so are ideal for bagging (which reduces the variance).

3. *Both bagging and boosting produce predictions by creating an ensemble of functions. Explain how these ensembles differ (you do not need to describe the methods for arriving at the ensembles just how structurally the ensembles will differ).*

   Bagging considers each sample to be of equal weight but each learner only sees a subset of the data. The bias of each learner is the same as the ensemble. Our motivation is variance reduction.

   Boosting places more weight on samples that previous models got wrong. Each learner sees all of the data but with variable sample weights. Learners are added an a non i.d. way to remove bias.

2 Problems – 2

1. *The Adaboost initialisation and update.*

   (a) *Explain how Adaboost chooses its initial weighting $D_0(i)$ of the examples.*

   $$S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\}.$$

   It chooses an equal weighting for all examples, i.e. $D_0(i) = 1/m$ for all i.

   (b) *Explain how it updates the distribution D_{t-1} after choosing a weak learner h_t with coefficient α_t at stage t including the normalising constant $Z(t)$.*

   The update is given by

   $$D_{t+1}(i) = \frac{D_t(i)e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

   where $\alpha_t := \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$, $\epsilon_t := \sum_{i=1}^m D_t(i)\mathcal{I}[h_t(\boldsymbol{x}_i) \neq y_i]$ and $Z_t := \sum_{i=1}^m D_t(i)e^{-\alpha_t y_i h_t(x_i)}$.

   (c) *Give an intuitive justification for this update.*

   If we have a correctly classified example, $y_i = h_t(\boldsymbol{x}_i)$, then $D_{t+1}(i) \propto D_t(i)e^{-\alpha_t}$. Since $\alpha_t \geq 0$, the weight for that example is reduced. Else, $y_i \neq h_t(\boldsymbol{x}_i)$, then $D_{t+1}(i) \propto D_t(i)e^{\alpha_t}$, i.e. the weight for that example is increased.

2. *Explain how Adaboost uses the distribution D_t that weights the examples $((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m))$ to choose a weak learner from a set H at stage t.*

   Any hypothesis that satisfies $\mathcal{E}_t := \sum_{i=1}^m D_t(i)\mathcal{I}[h_t(\boldsymbol{x}_i) \neq y_i] < \frac{1}{2}$ will work. However, the optimal weak learner is the hypothesis that minimises $\mathcal{E}_t$.

3. *Using the results of part (1.b), obtain a bound for*

   $$\frac{1}{m}\sum_{i=1}^m \mathcal{I}[\text{sign}(f_T(\boldsymbol{x}_i)) \neq y_i] \leq \frac{1}{m}\sum_{i=1}^m \exp(-y_i f_T(\boldsymbol{x}_i)),$$

   in terms of the normalisation constants $Z(1), \ldots, Z(T)$ where $\mathcal{I}[x]$ is the indicator function with value 1 when x is true and 0 otherwise, and $f_T(\boldsymbol{x}) = \sum_{i=1}^T \alpha_t h_t(\boldsymbol{x})$.

From part (1.b) we have that

$$D_{t+1}(i) = \frac{D_t(i)e^{-\alpha_t y_i h_t(x_i)}}{Z_t}.$$

This gives

$$D_{T+1}(i) = \frac{1}{m}\frac{\prod_{t=1}^{T} e^{-\alpha_t y_i h_t(x_i)}}{\prod_{t=1}^{T} Z_t}$$

We can thus write

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m}\exp(-y_i f_T(\boldsymbol{x}_i)) &= \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}_i)) \\
&= \frac{1}{m}\sum_{i=1}^{m}\prod_{t=1}^{T}\exp(-y_i\alpha_t h_t(\boldsymbol{x}_i)) \\
&= \sum_{i=1}^{m} D_{T+1}(i)\prod_{t=1}^{T} Z_t \\
&= \prod_{t=1}^{T} Z_t.
\end{aligned}$$

3 Question 3

A dataset $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ is generic iff $\mathbf{x}_i = \mathbf{x}_j \implies y_i = y_j$. A set of functions $\mathcal{H}$ is called weakly-universal if for every generic data set and weighting of the data $\{D_i\}_{i=1}^m$ such that $\sum_{i=1}^m D_i = 1$, there exists a function $h \in \mathcal{H}$ with weighted error $\sum_{i=1}^m D_i\,[h(\mathbf{x}_i) \neq y_i] < 0.5$.

3.a

We consider the set of all decision stumps

$$\mathcal{H}_{ds}^n \coloneqq \{s\, h_{i,z} : s \in \{-1, 1\}, i \in \{1, \ldots, n\}, z \in \mathbb{R}\}$$

where $h_{i,z} : \mathbb{R}^n \to \{-1, 1\}$ is such that for all $\mathbf{x} \in \mathbb{R}^n$

$$h_{i,z}(\mathbf{x}) \coloneqq \begin{cases} 1 & \text{if } x_i \leq z, \\ -1 & \text{if } x_i > z. \end{cases}$$

We prove that for $\mathcal{H}_{ds}^1$ is weakly-universal and that $\mathcal{H}_{ds}^n$ is not weakly-universal for $n > 1$.

Consider the case of $n = 1$. Suppose we have a generic dataset $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R} \times \{-1, 1\}$ and some weighting of the data $\{D_i\}_{i=1}^m$ such that $\sum_{i=1}^m D_i = 1$. By relabelling the dataset if necessary, we can assume without loss of generality that $x_1 \leq x_2 \leq \cdots \leq x_m$. Since the dataset is generic, if $\mathbf{x}_i = \mathbf{x}_j$ then $y_i = y_j$, so we can simply consider the sample only once and assign it a weight of $D_i + D_j$. Hence, without loss of generality, we can assume that all the $\{x_i\}_{i=1}^m$ are distinct and so that $x_1 < x_2 < \cdots < x_m$. Moreover, by removing data points if necessary, we can assume without loss of generality that $D_i > 0$ for $i = 1, \ldots, m$. Take $z = x_m$, so that $h_{1,x_m} \in \mathcal{H}_{ds}^1$ predicts 1 for each of $x_1, \ldots, x_m$. If the weighted error for h_{1,x_m}

$$\sum_{i=1}^{m} D_i\,[h_{1,x_m}(x_i) \neq y_i] = \sum_{i=1}^{m} D_i\,[1 \neq y_i]$$

is strictly smaller than 0.5 then we are done. If the weighted error for h_{1,x_m} is strictly greater than 0.5 then the weighted error for $-h_{1,x_m} \in \mathcal{H}^1_{ds}$ is

$$\sum_{i=1}^{m} D_i \left[-h_{1,x_m}(x_i) \neq y_i\right] = \sum_{i=1}^{m} D_i \left[-1 \neq y_i\right] = \sum_{i=1}^{m} D_i - \sum_{i=1}^{m} D_i \left[1 \neq y_i\right] = 1 - \sum_{i=1}^{m} D_i \left[h_{1,x_m}(x_i) \neq y_i\right] < 0.5$$

and we are done. So, assume that the weighted error for h_{1,x_m} is equal to 0.5. Recalling that $x_{m-1} < x_m$, we take $z = \frac{x_{m-1}+x_m}{2}$ so that $x_1 < x_2 < \cdots < x_{m-1} < z < x_m$. Hence, $h_{1,z} \in \mathcal{H}^1_{ds}$ predicts 1 for each of $x_1, \ldots, x_{m-1}$ and -1 for x_m. We get that the weighted error for $h_{1,z}$ is

$$\begin{aligned}
\sum_{i=1}^{m} D_i \left[h_{1,z}(x_i) \neq y_i\right] &= \sum_{i=1}^{m-1} D_i \left[1 \neq y_i\right] + D_m \left[-1 \neq y_m\right] \\
&= \sum_{i=1}^{m} D_i \left[1 \neq y_i\right] - D_m \left[1 \neq y_m\right] + D_m \left[-1 \neq y_m\right] \\
&= 0.5 + D_m \Big(\left[-1 \neq y_m\right] - \left[1 \neq y_m\right] \Big) \\
&= \begin{cases} 0.5 - D_m & \text{if } y_m = -1, \\ 0.5 + D_m & \text{if } y_m = 1. \end{cases}
\end{aligned}$$

Recall that $D_m > 0$. If $y_m = -1$ then $h_{1,z} \in \mathcal{H}^1_{ds}$ has weighted error strictly smaller than 0.5. If $y_m = 1$ then $h_{1,z} \in \mathcal{H}^1_{ds}$ has weighted error strictly greater than 0.5 and so, with the same reasoning as above, $-h_{1,z} \in \mathcal{H}^1_{ds}$ has weighted error strictly smaller than 0.5. We have proved that given any dataset and any weighting of it, we can find a function in $\mathcal{H}^1_{ds}$ with weighted error strictly smaller than 0.5, hence $\mathcal{H}^1_{ds}$ is weakly-universal.

Now, consider the case of $n > 1$. We provide a counter example to prove that $\mathcal{H}^n_{ds}$ is not weakly-universal. Our dataset in $\mathbb{R}^n \times \{-1, 1\}$ consists of

$$\begin{aligned}
\mathbf{x}_{0,0} &\coloneqq (0, 0, 0, \ldots, 0) \in \mathbb{R}^n \text{ with label } y_{0,0} \coloneqq -1, \\
\mathbf{x}_{0,1} &\coloneqq (0, 1, 0, \ldots, 0) \in \mathbb{R}^n \text{ with label } y_{0,1} \coloneqq 1, \\
\mathbf{x}_{1,0} &\coloneqq (1, 0, 0, \ldots, 0) \in \mathbb{R}^n \text{ with label } y_{1,0} \coloneqq 1, \\
\mathbf{x}_{1,1} &\coloneqq (1, 1, 0, \ldots, 0) \in \mathbb{R}^n \text{ with label } y_{1,1} \coloneqq -1.
\end{aligned}$$

with uniform weighting $D_{0,0} = D_{0,1} = D_{1,0} = D_{1,1} \coloneqq 0.25$. We need to show that for every $h \in \mathcal{H}^n_{ds}$, the weighted error

$$\sum_{i=0}^{1} \sum_{j=0}^{1} D_{i,j}[h(\mathbf{x}_{i,j}) = y_{i,j}] = 0.25 \sum_{i=0}^{1} \sum_{j=0}^{1} [h(\mathbf{x}_{i,j}) = y_{i,j}]$$

is exactly equal to 0.5. From this equation, we clearly see that the weighted error is exactly equal to 0.5 if and only if 2 points are correctly classified and 2 points are wrongly classified so that $0.25 \cdot 2 = 0.5$. Hence, we show by considering all cases that every $h \in \mathcal{H}^n_{ds}$ classifies 2 points correctly and 2 points wrongly.

For $i \in \{1, 2\}$ and $z \in (-\infty, 0) \cup [1, \infty)$, $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies all four points with the same labels, since the true labels are $\{-1, 1, 1, -1\}$ it must classify 2 correctly and 2 wrongly.

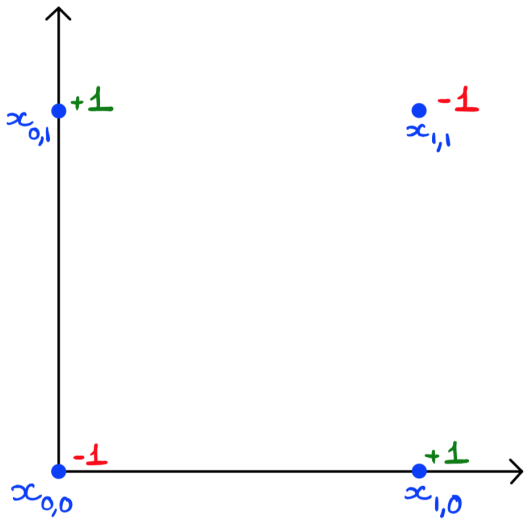
For $i = 1$ and $z \in [0, 1)$, $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies the 2 points $\mathbf{x}_{0,0}, \mathbf{x}_{0,1}$ with the same label and the 2 other points $\mathbf{x}_{1,0}, \mathbf{x}_{1,1}$ with the other label. Since $\mathbf{x}_{0,0}, \mathbf{x}_{0,1}$ have true labels $-1, 1$ and $\mathbf{x}_{1,0}, \mathbf{x}_{1,1}$ have true labels $1, -1$, we deduce that $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies 2 correctly and 2 wrongly.

Similarly, for $i = 2$ and $z \in [0, 1)$, $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies the 2 points $\mathbf{x}_{0,0}, \mathbf{x}_{1,0}$ with the same label and the 2 other points $\mathbf{x}_{0,1}, \mathbf{x}_{1,1}$ with the other label. Since $\mathbf{x}_{0,0}, \mathbf{x}_{1,0}$ have true labels $-1, 1$ and $\mathbf{x}_{0,1}, \mathbf{x}_{1,1}$ have true labels $1, -1$, we deduce that $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies 2 correctly and 2 wrongly.

If $n > 2$, for $i > 2$ and $z \in \mathbb{R}$, $\pm h_{i,z} \in \mathcal{H}^n_{ds}$ classifies all four points with the same labels, since the true labels are $\{-1, 1, 1, -1\}$ it must classify 2 correctly and 2 wrongly.

We have proved that every $h \in \mathcal{H}_{ds}^n$ classifies 2 points correctly and 2 points wrongly, and hence that every $h \in \mathcal{H}_{ds}^n$ has a weighted error of exactly 0.5. We have proved that $h \in \mathcal{H}_{ds}^n$ is not weakly-universal for $n > 1$.

Geometrically, we can plot the points projected in $\mathbb{R}^2$ and clearly see that any line perpendicular to one of the axes will either separate all four points on one side and none on the other, or separate two points with true labels $\{-1, 1\}$ on one side and two other points also with true labels $\{-1, 1\}$ on the other side.



3.b

Let $\mathcal{S}$ denote the set of all strings of non-zero finite length over $\{a, b\}$. We consider the set of all substring-match functions

$$\mathcal{H}_{\mathcal{SS}} \coloneqq \{s\, g_z : s \in \{-1, 1\}, z \in \mathcal{S}\}$$

where $g_z : \mathcal{S} \to \{-1, 1\}$ is such that for all $x \in \mathcal{S}$

$$g_z(x) \coloneqq \begin{cases} 1 & \text{if } x \sqsubseteq z, \\ -1 & \text{if } x \not\sqsubseteq z. \end{cases}$$

We prove that $\mathcal{H}_{\mathcal{SS}}$ is weakly-universal. Suppose we have a generic dataset $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{S} \times \{-1, 1\}$ and some weighting of the data $\{D_i\}_{i=1}^m$ such that $\sum_{i=1}^m D_i = 1$. Since the dataset is generic, if $\mathbf{x}_i = \mathbf{x}_j$ then $y_i = y_j$, so we can simply consider the sample only once and assign it a weight of $D_i + D_j$. Hence, without loss of generality, we can assume that all the $\{x_i\}_{i=1}^m$ are distinct. By removing data points if necessary, we can also assume without loss of generality that $D_i > 0$ for $i = 1, \ldots, m$.

Take $z = x_1 \cdots x_m \in \mathcal{H}_{\mathcal{SS}}$, that is, the string obtained by concatenating all the strings in our dataset. Since $x_i \sqsubseteq z$ for $i = 1, \ldots, m$, we deduce that $h_z \in \mathcal{H}_{\mathcal{SS}}$ predicts 1 for each of $x_1, \ldots, x_m$, and so the weighted error for h_z is

$$\sum_{i=1}^m D_i \left[h_z(x_i) \neq y_i\right] = \sum_{i=1}^m D_i \left[1 \neq y_i\right].$$

If this is strictly smaller than 0.5 then we are done. If the weighted error for h_z is strictly greater than 0.5, then the weighted error for $-h_z \in \mathcal{H}_{\mathcal{SS}}$ is

$$\sum_{i=1}^m D_i \left[-h_z(x_i) \neq y_i\right] = \sum_{i=1}^m D_i \left[-1 \neq y_i\right] = \sum_{i=1}^m D_i - \sum_{i=1}^m D_i \left[1 \neq y_i\right] = 1 - \sum_{i=1}^m D_i \left[h_z(x_i) \neq y_i\right] < 0.5$$

and we are done. So, suppose that the weighted error for h_z is equal to 0.5. Let $k \in \{1, \ldots, m\}$ be such that x_k is an element of $x_1, \ldots, x_m$ with maximal length (k might not be unique). Since our dataset

$x_1, \ldots, x_m$ consists of distinct strings, we must have $x_k \not\sqsubseteq x_i$ for $i \neq k$ because x_i either has smaller length than x_k or has the same length but is different. Now, consider $z = x_1 \cdots x_{k-1} x_{k+1} \cdots x_m$, that is, the string obtained by concatenating all the strings in our dataset except x_k. Then, we have $x_i \sqsubseteq z$ for $i \neq k$ and we want to have '$x_k \not\sqsubseteq z$' but this might not always be the case as maybe from concatenating some strings we could have formed the same consecutive sequence of characters as in x_k. To solve this, we can consider different permutations of $\{x_i : i = 1, \ldots, k-1, k+1, \ldots m\}$ before concatenating them. If this still does not solve our problem, then we can add extra strings from $\mathcal{S}$ in between our $x_1, \cdots, x_{k-1}, x_{k+1}, \cdots, x_m$ in the concatenation so that we indeed have $x_k \not\sqsubseteq z$. By choosing the right extra strings from $\mathcal{S}$, we can always construct z such that $x_k \not\sqsubseteq z$ and $x_i \sqsubseteq z$ for $i \neq k$. So, the weighted error of $h_z \in \mathcal{H}_{\mathcal{SS}}$ is

$$\begin{aligned}
\sum_{i=1}^{m} D_i \left[h_z(x_i) \neq y_i\right] &= \sum_{i \neq k} D_i \left[h_z(x_i) \neq y_i\right] + D_k \left[h_z(x_k) \neq y_k\right] \\
&= \sum_{i \neq k} D_i \left[1 \neq y_i\right] + D_k \left[-1 \neq y_k\right] \\
&= \sum_{i=1}^{m} D_i \left[1 \neq y_i\right] - D_k \left[1 \neq y_k\right] + D_k \left[-1 \neq y_k\right] \\
&= 0.5 + D_k \Big(\left[-1 \neq y_k\right] - \left[1 \neq y_k\right] \Big) \\
&= \begin{cases} 0.5 - D_k & \text{if } y_k = -1, \\ 0.5 + D_k & \text{if } y_k = 1. \end{cases}
\end{aligned}$$

Since $D_k > 0$, we find that the weighted error of $h_z \in \mathcal{H}_{\mathcal{SS}}$ is either strictly greater or strictly smaller than 0.5. If it is strictly greater than 0.5, then the weighted error of $-h_z \in \mathcal{H}_{\mathcal{SS}}$ is strictly smaller than 0.5 with the same proof as above. Hence, we have proved that for any generic dataset and any weighting of it, there exists a function $h \in \mathcal{H}_{\mathcal{SS}}$ with weighted error strictly smaller than 0.5 and so that $\mathcal{H}_{\mathcal{SS}}$ is weakly-universal.

Note that the structure of this proof is exactly the same as the one for the proof in 3.a of $\mathcal{H}^1_{ds}$ being weakly-universal. This is because the result holds more generally for some partially ordered sets, and in particular holds for $(\mathbb{R}, \leq)$ and for $(\mathcal{S}, \sqsubseteq)$.

Lecture 5 – Questions

Lisa Tse

December 3, 2021

1 Problems – 1

1. *Suppose $\mathcal{X} = \{True, False\}^n$. Give a polynomial time algorithm A with a mistake bound $\mathcal{M}(A) \leq O(n^2)$ for any example sequence which is consistent with a k -literal conjunction. Your answer should contain an argument that $\mathcal{M}(A) \leq O(n^2)$.*

   We take the perceptron algorithm for simplicity. We have the following mistake bound:

   $$\mathcal{M} \leq R^2 ||\boldsymbol{u}||^2$$

   where $R^2 = \max_t ||\boldsymbol{x}_t||^2$, and this holds for any $\boldsymbol{u}$ for which $\langle \boldsymbol{u}, \boldsymbol{x}_t \rangle y_t \geq 1$. To apply this to the conjunction problem, we make use of de Morgan's Law, which states that $x \wedge y = \overline{(\bar{x} \vee \bar{y})}$, and represent the conjunction as a disjunction of the negated variables. We take S to be the set of indices of the variables in the conjunction, and define

   $$u_i = \begin{cases} -2 & \text{if } i \in S \\ 1 & \text{if } i = n+1 \\ 0 & \text{otherwise .} \end{cases}$$

   Instead of considering $\boldsymbol{x} \in \{0,1\}^n$, we take the feature map $\phi(\boldsymbol{x}) = (\bar{\boldsymbol{x}}, 1)$, where $\bar{x}_i = \begin{cases} 1 & \text{if } x_i = 0 \\ 0 & \text{if } x_i = 1 \end{cases}$. Then we have that $R^2 \leq n+1$, and $||\boldsymbol{u}||^2 = 4k+1$, giving

   $$\mathcal{M}(A) \leq (n+1)(4k+1).$$

2. *State the perceptron convergence theorem [Novikoff] explaining the relation with the hard margin support vector machine solution*

   The number of mistakes made by the perceptron algorithm is:

   $$\mathcal{M} \leq \frac{R^2}{\gamma^2},$$

   if there exists a vector $\boldsymbol{v}$ that satisfy $\langle \boldsymbol{v}, \boldsymbol{x}_t \rangle y_t \geq \gamma$ and $||\boldsymbol{v}|| = 1$. The γ term is analogous to the SVM margin term.

3. *Define the c -regret of learning algorithm as*

$$c - regret(m) = L_A(S) - c \min_{i \in [n]} L_i(S)$$

thus the usual regret is the 1-regret.

(a) *Argue for the weighted majority set-up argue that without randomised prediction it is impossible for all training sequences to obtain sublinear c-regret for $c < 2$.*

We consider a training sequence with two experts E_1 and E_2 that give constant predictions, so that for all $t \in [T]$

$$x_{t,1} = 0$$

$$x_{t,2} = 1$$

We also consider an adversarial environment that forces mistakes for each prediction. Then, $\min_{i \in \{1,2\}} \mathcal{M}_i \leq \frac{T}{2}$, but $\mathcal{M}_A = T$. Hence we have that $\mathcal{M}_A \geq 2 \min_{i \in \{1,2\}} \mathcal{M}_i$

(b) *Show how to select β to obtain sublinear 2-regret.* Starting with

$$\mathcal{M} \leq \frac{\ln(\frac{1}{\beta})}{\ln(\frac{2}{1+\beta})} M_i + \frac{1}{\ln(\frac{2}{1+\beta})} \ln(n)$$

Defining $\epsilon := 1 - \beta$, and assuming that $\epsilon \in [0, 1/2]$ so that we can use the inequality $-\log(1-\epsilon) \leq \epsilon + \epsilon^2$ for $\epsilon \in [0, 1/2]$, we obtain

$$\ln\left(\frac{1}{\beta}\right) = -\ln(1-\epsilon) \leq \epsilon + \epsilon^2$$

and using the inequality $-\log(1+x) \geq -x$

$$\ln\left(\frac{2}{1+\beta}\right) = -\ln\left(\frac{1+\beta}{2}\right) = -\ln\left(1 + \frac{\epsilon}{2}\right) \geq \frac{\epsilon}{2}$$

thus giving

$$\begin{aligned} \mathcal{M} &\leq 2(1+\epsilon)\mathcal{M}_i + \frac{2}{\epsilon} \ln(n) \\ &\leq 2\mathcal{M}_i + 2\epsilon T + \frac{2}{\epsilon} \ln(n) \end{aligned}$$

Tuning $\epsilon = \sqrt{\frac{\log(n)}{T}}$, and assuming that $\sqrt{\frac{\log(n)}{T}} \leq \frac{1}{2}$ we then

$$\mathcal{M} \leq 2M_i + 4\sqrt{\log(n)T}$$

In the case that $\sqrt{\frac{\log(n)}{T}} > \frac{1}{2}$, we can write the generic bound

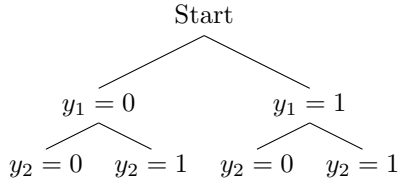
$$\begin{aligned}\mathcal{M} &\leq T \\ &\leq 2\mathcal{M}_i + T \\ &< 2\mathcal{M}_i + 4\log(n)\end{aligned}$$

Combining the 2 cases, we then have

$$\mathcal{M} \leq 2M_i + 4\sqrt{\log(n)T} + 4\log(n).$$

4. *Consider binary prediction with expert advice, with a perfect expert. Prove that any algorithm makes at least* $\Omega(\min(T, \log_2 n))$ *mistakes in the worst case.*

   Let's assume that $\log_2 n$ is an integer. Consider the case where we have experts that give each of the following predictions in the tree:



   Defining N to be the tree-depth $- 1$, we have 2^N experts, and N mistakes can be forced when $N \geq T$, and T mistakes if $N < T$. Taking $n = 2^N$, we then have $\min(T, \log_2(n))$ forced mistakes.

2 Problems − 2

1. *Recall that by tuning the weighted majority we achieved a bound*

$$\mathcal{M} \leq 2.63 \min_i \mathcal{M}_i + 2.63 \ln n.$$

   Now by using randomisation in the prediction, design an algorithm with a bound that has the property

$$\frac{\mathcal{M}}{T} \leq \min_{i \in [n]} \frac{\mathcal{M}_i}{T}$$

   as $T \to \infty$, for the weighted majority setting (i.e., the mean the prediction error of the algorithm is bounded by the mean prediction error of the "best" expert). Recalling that m is the number of examples (and the "tuning" of the algorithm may depend on m). For contrast compare this to problem 3.1 above.

   Initialise weights of all experts to be 1. We consider the following protocol at each timestep t:

(a) Receive expert predictions $\boldsymbol{x}_t \in \{0,1\}^n$

(b) Predict $\hat{y}_t = \begin{cases} 0 & \text{with probability } \frac{\sum_{i:x_{t,i}=0} w_{t.i}}{W_t} \\ 1 & \text{otherwise} \end{cases}$.

(c) Receive $y_t \in \{0,1\}$

(d) Weights of incorrect experts multiplied by $\beta \in [0,1)$.

Define A_t to be the proportion of experts that give the wrong prediction at time t. Note that the $\mathbb{E}[\mathcal{M}] = \sum_{t=1}^{T} A_t$. We have

$$\begin{aligned} W_{T+1} &= W_T((1-A_T) + A_T\beta) \\ &= W_T(1-(1-\beta)A_T) \\ &= W_1 \prod_{t=1}^{T}(1-(1-\beta)A_t) \\ &\leq n \prod_{t=1}^{T} \exp(-(1-\beta)A_t) \\ &= n \exp(-\sum_{t=1}^{T}(1-\beta)A_t) \\ &= n \exp(-(1-\beta)\mathbb{E}[\mathcal{M}]) \end{aligned}$$

where the inequality comes from $\exp(-x) \geq 1-x$. Since we also have that $W_{T+1} = \sum_{i=1}^{n} \beta^{M_i} \geq \beta^{M_i}$ for any $i \in [n]$, we have

$$\beta^{M_i} \leq n \, \exp(-(1-\beta)\mathbb{E}[\mathcal{M}]).$$

Taking logs, we obtain

$$M_i \log(\beta) \leq \log(n) - (1-\beta)\mathbb{E}[\mathcal{M}].$$

Rearranging,

$$\mathbb{E}[\mathcal{M}] \leq \frac{-\log\beta}{1-\beta} M_i + \frac{1}{1-\beta}\log(n)$$

Defining $\epsilon := 1-\beta$ and assuming that $\epsilon \in [0,1/2]$ so that we can use the inequality $-\log(1-\epsilon) \leq \epsilon + \epsilon^2$ for $\epsilon \in [0,1/2]$, the following follows

$$\begin{aligned} \mathbb{E}[\mathcal{M}] &\leq (1+\epsilon)M_i + \frac{1}{\epsilon}\log(n) \quad (1) \\ &\leq M_i + \epsilon T + \frac{1}{\epsilon}\log(n) \quad (2) \end{aligned}$$

Then tune $\epsilon = \sqrt{\frac{\log(n)}{T}}$ which is in [0,1/2] for $T \geq \log(n)$, thus giving

$$\mathbb{E}[\mathcal{M}] - M_i \leq 2\sqrt{\log(n)T}.$$

$$\frac{\mathbb{E}[\mathcal{M}]}{T} - \frac{M_i}{T} \leq 2\sqrt{\frac{\log(n)}{T}}.$$

The upper bound vanishes as $T \to \infty$.

Lecture 6 – Questions

Shota Saito

December 15, 2021

1 Problems – 1

1.1 Problem 1.1

Unsupervised learning learns how the models can infer a function to describe the hidden structure of the given unlabeled dataset. In supervised learning, the model learns from the labeled data and predict the label of unseen data. Semi-supervised learning is somewhere between unsupervised learning and supervised learning. In semi-supervised learning, the models infer a function to describe the hidden structure of the data, but using unlabeled data and the labeled data at the same time.

1.2 Problem 1.2

We label the indices to the five points as in the Fig. 1. In this problem, we draw a unweighted and undirected graph. If one vertex are a neighbor of another vertex, we draw a undirected edge. We list the 2 nearest neighbors as follows.

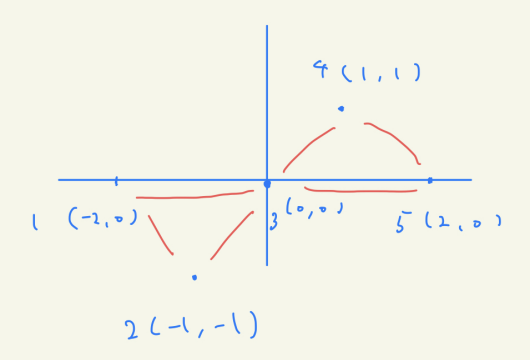


Figure 1: The data for problem 1.2. Red line shows the edge drawn by 2-nearest neighbors.

1. 2, 3
2. 1, 3
3. 2, 4
4. 3, 5
5. 3, 4

Therefore, we can draw red lines and formulate a graph in Fig. 1.

1.3 Problem 1.3

From the graph in Fig. 1, we obtain the adjacency matrix for this graph A as

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Also, the degree matrix D is given as

$$D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}.$$

Therefore, the graph Laplacian L can be written as

$$\begin{aligned} L &= D - A \\ &= \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{pmatrix}. \end{aligned}$$

1.4 Problem 1.4

1.4.1 i

For the minimum cut of this graph, the label for v_3 can take either 1 or -1. In the minimum cut setting, v_1 should be 1. However, v_3 and v_4 take $(1,1), (1,-1)$, and $(-1,1)$, all of which yield the minimum cut 1.

1.4.2 ii

In order to minimize the energy S_2, which is defined as

$$S_2 := \sum_{(u,v)\in E} |\psi(u) - \psi(v)|^2, \tag{1}$$

where E is a set of edges of the graph. Clearly, the energy is minimized when $\psi(v_1) = 1$. Also, the energy is minimized when

$$|\psi(v_2) - \psi(v_3)| = |\psi(v_3) - \psi(v_4)| = |\psi(v_4) - \psi(v_5)|. \tag{2}$$

Therefore, we obtain

$$\psi(v_3) = 1/3. \tag{3}$$

2 Problem 2

We label the indices to the vertices of D_m as in Fig. 2. Define the energy S_2 as

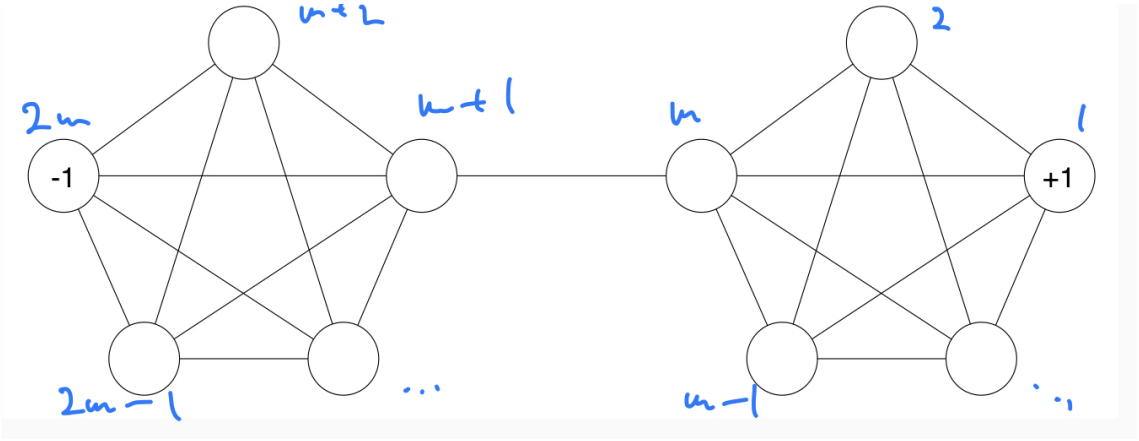


Figure 2: The illustration of graph D_m when $m = 5$. We also attach the indices to the vertices as above.

$$S_2 := \sum_{(u,v)\in E} |\psi(u) - \psi(v)|^2. \tag{4}$$

The harmonic function is obtained for all $i \in [2m]$ when

$$\frac{\partial S_2}{\partial \psi(v_i)} = 0. \tag{5}$$

From symmetric structure of graph, we obtain the following

$$\begin{aligned}
\psi(v_2) &= \ldots = \psi(v_{m-1}) \\
\psi(v_{m+2}) &= \ldots = \psi(v_{2m-1}) \\
\psi(v_m) &= -\psi(v_{m+1}) \\
\psi(v_2) &= -\psi(v_{m+2})
\end{aligned} \quad .$$

Then, Eq. (5) can be further written as

$$\frac{\partial S_2}{\partial \psi(v_j)} = -\psi(v_m) + 2\psi(v_j) - \psi(v_1) \text{ for } j = 2, \ldots, m-1 \tag{6}$$

$$\frac{\partial S_2}{\partial \psi(v_j)} = (\psi(v_j) - \psi(v_1)) + \sum_{i=2}^{m-1} (\psi(v_i) - \psi(v_m)) + (\psi(v_j) - \psi(v_{m+1})) \text{ for } j = m \tag{7}$$

$$\frac{\partial S_2}{\partial \psi(v_j)} = (\psi(v_j) - \psi(v_m)) + \sum_{i=m+1}^{2m-1} (\psi(v_j) - \psi(v_i)) + (\psi(v_j) - \psi(v_{2m})) \text{ for } j = m+1 \tag{8}$$

$$\frac{\partial S_2}{\partial \psi(v_j)} = -\psi(v_m) + 2\psi(v_j) - \psi(v_1) \text{ for } j = m+2, \ldots, 2m-1 \tag{9}$$

From Eq. (6) and Eq. (7), we obtain

$$\begin{aligned}
\psi(v_j) &= \frac{m+2}{m+4} \text{ for } j = 2, \ldots, m-1, \\
\psi(v_j) &= \frac{m}{m+4} \text{ for } j = m, \\
\psi(v_j) &= -\frac{m}{m+4} \text{ for } j = m+1, \\
\psi(v_j) &= -\frac{m+2}{m+4} \text{ for } j = m+1, \ldots, 2m-1.
\end{aligned}$$

Therefore, in the asymptotic case when $m \to \infty$,

$$\begin{aligned}
\psi(v_j) &\to 1 \text{ for } j = 2, \ldots, m, \\
\psi(v_j) &\to -1 \text{ for } j = m+1, \ldots, 2m.
\end{aligned}$$

3 Problem 3

3.1 i

Let L be a graph Laplacian, and L^+ be a psuedoinverse of graph Laplacian L. From the lecture slide, we have

$$r_G(i,j) = \mathbf{e}_i L^+ \mathbf{e}_i - 2\mathbf{e}_i L^+ \mathbf{e}_j + \mathbf{e}_j L^+ \mathbf{e}_j. \tag{10}$$

Since L^+ is pseudoinverse of L, the following holds;

$$L^+ = (L^\top L)^+ L^\top = (L^2)^+ L.$$

Therefore, we obtain

$$\sum_{j=1}^{n} L^+ \mathbf{e}_j = (L^2)^+ L\mathbf{1} = \mathbf{0}. \tag{11}$$

Exploiting Eq. (11), we compute

$$\begin{aligned}\sum_{j=1}^{n} \frac{1}{n} r_g(i,j) &= \frac{1}{n}\sum_{j=1}^{n} (\mathbf{e}_i L^+ \mathbf{e}_i - 2\mathbf{e}_i L^+ \mathbf{e}_j + \mathbf{e}_j L^+ \mathbf{e}_j) \\ &= \mathbf{e}_i L^+ \mathbf{e}_i + \frac{1}{n}\sum_{j=1}^{n} \mathbf{e}_j L^+ \mathbf{e}_j. \end{aligned} \tag{12}$$

This means that the following inequality holds;

$$L^+_{ii} \leq \frac{1}{n}\sum_{j=1}^{n} r_G(i,j). \tag{13}$$

If we take max of this inequality, we can obtain the desired inequality.

Supervised Learning
Learning Theory

Antonin Schrab

Problem 1. (problem 3.3 in [SS14])

Let $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{0,1\}$, and let $\mathcal{H}$ be the class of concentric circles in the plane, that is, $\mathcal{H} = \{h_r : r > 0\}$, where $h_r(\mathbf{x}) = \mathbb{1}(\|\mathbf{x}\|_2 \leq r)$. Prove that $\mathcal{H}$ is PAC learnable (assume realizability), and its sample complexity is bounded by

$$m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{\ln(1/\delta)}{\epsilon} \right\rceil.$$

We need to show that there exists a learning algorithm $\mathcal{A}$ with the following property. For all $\epsilon, \delta \in (0,1)$, all distribution $\mathcal{D}$ over $\mathbb{R}^2$ and every labelling function $f^* : \mathbb{R}^2 \to \{0,1\}$, if the realizable assumption holds, then when running the learning algorithm on $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ where $\mathbf{x}_i \overset{iid}{\sim} \mathcal{D}$ and $y_i \coloneqq f^*(\mathbf{x}_i)$ for $i = 1, \dots, m$ where $m \geq \left\lceil \frac{\ln(1/\delta)}{\epsilon} \right\rceil$, the algorithm returns a hypothesis $h = \mathcal{A}(S)$ such that $L_{(\mathcal{D}, f^*)}(h) = \mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(h(\mathbf{x}) \neq f^*(\mathbf{x})\big) \leq \epsilon$ holds with probability of at least $1 - \delta$ with respect to the randomness of S. This condition can be rewritten as

$$\mathbb{P}_{S\sim D^m}\left(\mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big([\mathcal{A}(S)](\mathbf{x}) \neq f^*(\mathbf{x})\big) \leq \epsilon\right) \leq 1 - \delta.$$

Let $\mathcal{D}$ be a distribution over $\mathbb{R}^2$ and let $f^* : \mathbb{R}^2 \to \{0,1\}$ be a labelling function. We assume realizability, that is $\mathcal{D}$ and f^* are such that there exists some $r^\star > 0$ such that $h_{r^\star} \in \mathcal{H}$ satisfies

$$0 = L_{\mathcal{D}}(h_{r^\star}) = \mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(h_{r^\star}(\mathbf{x}) \neq f^*(\mathbf{x})\big) = \mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(\mathbb{1}(\|\mathbf{x}\|_2 \leq r^\star) \neq f^*(\mathbf{x})\big),$$

so we must have

$$f^*(\mathbf{x}) = \mathbb{1}(\|\mathbf{x}\|_2 \leq r^\star) = \begin{cases} 1 & \text{if } \|\mathbf{x}\|_2 \leq r^\star \\ 0 & \text{if } \|\mathbf{x}\|_2 > r^\star \end{cases} \tag{1}$$

for all $\mathbf{x} \in \mathbb{R}^2$.

We consider the algorithm which given some training sequence $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, where $\mathbf{x}_i \overset{iid}{\sim} \mathcal{D}$ and $y_i \coloneqq f^*(\mathbf{x}_i)$ for $i = 1, \dots, m$, returns h_{r_S} where $r_S \coloneqq \max_{i=1,\dots,m} \{\|\mathbf{x}_i\|_2 : y_i = 1\} > 0$. By Eq. (1), we must have $r_S \leq r^\star$.

Let $\epsilon \in (0,1)$. Define r_ϵ to be such that $\mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\left(r_\epsilon \leq \|\mathbf{x}\|_2 \leq r^\star\right) = \epsilon$. We then have

$$\begin{aligned}
\mathbb{P}_{S\sim\mathcal{D}^m}\left(L_{\mathcal{D}}(h_{r_S}) \geq \epsilon\right) &= \mathbb{P}_{S\sim\mathcal{D}^m}\left(\mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(h_{r_S}(\mathbf{x}) \neq f^*(\mathbf{x})\big) \geq \epsilon\right) \\
&= \mathbb{P}_{S\sim\mathcal{D}^m}\left(\mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(r_S \leq \|\mathbf{x}\|_2 \leq r^\star\big) \geq \epsilon\right) \\
&= \mathbb{P}_{S\sim\mathcal{D}^m}\left(\mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\big(r_S \leq \|\mathbf{x}\|_2 \leq r^\star\big) \geq \mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\left(r_\epsilon \leq \|\mathbf{x}\|_2 \leq r^\star\right)\right) \\
&= \mathbb{P}_{S\sim\mathcal{D}^m}\left(r_S \leq r_\epsilon\right) \\
&= \mathbb{P}_{S\sim\mathcal{D}^m}\left(\bigcap_{i=1}^m \{\|\mathbf{x}_i\|_2 \leq r_\epsilon\} \cup \{\|\mathbf{x}_i\|_2 \geq r^\star\}\right) \\
&= \prod_{i=1}^m \mathbb{P}_{\mathbf{x}_i\sim\mathcal{D}}\left(\{\|\mathbf{x}_i\|_2 \leq r_\epsilon\} \cup \{\|\mathbf{x}_i\|_2 \geq r^\star\}\right) \\
&= \left(1 - \mathbb{P}_{\mathbf{x}\sim\mathcal{D}}\left(r_\epsilon \leq \|\mathbf{x}\|_2 \leq r^\star\right)\right)^m \\
&= (1 - \epsilon)^m \\
&\leq e^{-\epsilon m} \leq \delta
\end{aligned}$$

for $\delta \in (0,1)$ satisfying

$$\delta \geq e^{-\epsilon m} \qquad \Longleftrightarrow \qquad m \geq \frac{\ln(1/\delta)}{\epsilon}.$$

We have used the fact that $1-\epsilon \leq e^{-\epsilon}$ which is proved in Problem 2.

Problem 2.

Prove $1-x \leq e^{-x}$ *for* $x \in \mathbb{R}$.

Note that

$$1-x \leq e^{-x} \text{ for all } x \in \mathbb{R} \qquad \Longleftrightarrow \qquad g(x) \coloneqq e^{-x}+x-1 \geq 0 \text{ for all } x \in \mathbb{R}.$$

So, we need to prove $g(x) \geq 0$ for all $x \in \mathbb{R}$. We have $g'(x) = -e^{-x}+1$ for all $x \in \mathbb{R}$ and $g''(x) = e^{-x} > 0$ for all $x \in \mathbb{R}$. Using the result on p10 of the Kernel slides, we deduce that g is convex. It attains its minimum when

$$0 = g'(x) = -e^{-x}+1 \qquad \Longleftrightarrow \qquad x = 0.$$

hence, we have

$$g(x) \geq g(0) = 0 \text{ for all } x \in \mathbb{R}.$$

We deduce that

$$1-x \leq e^{-x} \text{ for all } x \in \mathbb{R}.$$

Problem 3. (problem 5.3 in [SS14])

Prove that if $|\mathcal{X}| \geq km$ *for a positive integer* $k \geq 2$*, then we can replace the lower bound of* $1/4$ *in the No-Free-Lunch theorem with* $\frac{k-1}{2k} = \frac{1}{2} - \frac{1}{2k}$*. Namely, let* $\mathcal{A}$ *be a learning algorithm for the task of binary classification. Let* m *be any number smaller than* $|\mathcal{X}|/k$*, representing a training set size. Then, there exists a distribution* $\mathcal{D}$ *over* $\mathcal{X} \times \{0,1\}$ *such that:*

- *There exists a function* $f\colon \mathcal{X} \to \{0,1\}$ *with* $L_{\mathcal{D}}(f) = 0$.
- $\mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))] \geq \frac{1}{2} - \frac{1}{2k}$.

We refer the reader to the proof of the No-Free-Lunch Theorem (Theorem 5.1 in [SS14]). The statement of this result is presented on p47 of the Introduction slides and a sketch of the proof is provided on p48. The proof in in [SS14] holds a set $\mathcal{X}$ of size $2m$ and it is shown (Eq. (5.1)) that

$$\max_{i\in[T]} \mathbb{E}_{S\sim\mathcal{D}_i^m}[L_{\mathcal{D}_i}(A(S))] \geq 1/4.$$

The question asks us to generalise this result by showing that when $\mathcal{X}$ has size greater than km for some integer $k \geq 2$, we he have

$$\max_{i\in[T]} \mathbb{E}_{S\sim\mathcal{D}_i^m}[L_{\mathcal{D}_i}(A(S))] \geq \frac{k-1}{2k}.$$

Note that for $k = 2$, we recover $1/4$.

The reasoning of Eq. (5.3) and Eq. (5.4) still holds. Let C be a subset of $\mathcal{X}$ of size km. As in the proof, we let $v_1, \ldots, v_p$ be examples in C which do not appear in a set of size m, we

deduce that $p \geq (k-1)m$. Eq. (5.5), which used to be $L_{\mathcal{D}_i}(h) = \frac{1}{km}\sum_{x\in C}\mathbb{1}(h(x) \neq f_i(x)) \geq \frac{1}{2p}\sum_{r=1}^{p}\mathbb{1}(h(v_r) \neq f_i(v_r))$, then becomes

$$\begin{aligned} L_{\mathcal{D}_i}(h) &= \frac{1}{km}\sum_{x\in C}\mathbb{1}(h(x) \neq f_i(x)) \\ &\geq \frac{1}{km}\sum_{r=1}^{p}\mathbb{1}(h(v_r) \neq f_i(v_r)) \\ &\geq \frac{k-1}{kp}\sum_{r=1}^{p}\mathbb{1}(h(v_r) \neq f_i(v_r)). \end{aligned}$$

We observe that we have replaced the factor $\frac{1}{2}$ by the factor $\frac{k-1}{k}$. The rest of the proof with this new factor still holds with the exact same reasoning. Instead of obtaining

$$\max_{i\in[T]} \mathbb{E}_{S\sim\mathcal{D}_i^m}[L_{\mathcal{D}_i}(A(S))] \geq \frac{1}{4} = \frac{1}{2}\cdot\frac{1}{2},$$

we then obtain

$$\max_{i\in[T]} \mathbb{E}_{S\sim\mathcal{D}_i^m}[L_{\mathcal{D}_i}(A(S))] \geq \frac{1}{2}\cdot\frac{k-1}{k}$$

as required.

Problem 4. (problem 6.2 in [SS14])

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

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

1. We prove that $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) = \min\{k, |\mathcal{X}| - k\}$.

Consider a set $C \coloneqq \{x_1, \ldots, x_\ell\}$ consisting of $\ell \coloneqq \min\{k, |\mathcal{X}| - k\}$ elements of $\mathcal{X}$. If $\ell = k$ then $k \leq |\mathcal{X}| - k = |\mathcal{X}| - \ell$. If $\ell = |\mathcal{X}| - k$ then $|\mathcal{X}| - \ell = k$. So, in both cases, we have $|\mathcal{X}| - \ell \geq k$. Consider a labelling $\{y_1, \ldots, y_\ell\}$ with values in $\{0,1\}$. We have $N_+ \coloneqq \sum_{i=1}^{\ell} y_i \leq \ell \leq k$ elements with labels 1. We consider a classifier that predicts the correct labels on C. In order for this classifier to belong to $\mathcal{H}^{\mathcal{X}}_{=k}$, we require it to predict the label 1 on exactly $k - N_+$ other elements, that is, elements from the set $\mathcal{X} \setminus C$. Since the set $\mathcal{X} \setminus C$ has size $|\mathcal{X}| - \ell \geq k$, we can indeed select $k - N_+$ elements of it and predict 1 for them and 0 for the rest. The classifier we have constructed belongs to $\mathcal{H}^{\mathcal{X}}_{=k}$ and classifies elements of $C = \{x_1, \ldots, x_\ell\}$ correctly, so $\mathcal{H}^{\mathcal{X}}_{=k}$ shatters C. Hence, we have $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) \geq \min\{k, |\mathcal{X}| - k\}$

Consider a set of $k+1$ elements, all with labels 1. By definition of $\mathcal{H}^{\mathcal{X}}_{=k}$, there does not exist a classifier in $\mathcal{H}^{\mathcal{X}}_{=k}$ which classifies all those points correctly. We deduce that $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) < k+1$.

Consider a set of $|X| - k + 1$ elements, all with labels 0. Then, any classifier which classify those elements correctly can classify at most $k-1$ elements (the remaining ones) as 1 and so cannot belong to $\mathcal{H}^{\mathcal{X}}_{=k}$. We deduce that $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) < |X| - k + 1$.

We deduce that $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) < 1 + \min\{k, |\mathcal{X}| - k\}$ and conclude that $\mathrm{VCdim}(\mathcal{H}^{\mathcal{X}}_{=k}) = \min\{k, |\mathcal{X}| - k\}$.

2. We prove that $\mathrm{VCdim}(\mathcal{H}_{at-most-k}) = \min\{2k+1, |\mathcal{X}|\}$.

Suppose $2k+1 \leq |\mathcal{X}|$, we show that $\text{VCdim}(\mathcal{H}_{at-most-k}) \geq 2k+1$. Consider a set $C \coloneqq \{x_1, \ldots, x_{2k+1}\}$ consisting of $2k+1$ elements of $\mathcal{X}$. Consider a labelling $\{y_1, \ldots, y_{2k+1}\}$ with values in $\{0,1\}$. We have $N_+ \coloneqq \sum_{i=1}^{\ell} y_i$ elements of C with labels 1, and $N_- = 2k+1-N_+$ elements with labels 0. If $N_+ \leq k$, then the classifier, defined as $h(\mathbf{x}_i) \coloneqq y_i$ for $i = 1, \ldots, 2k+1$ and $h(\mathbf{x}) = 0$ for $\mathbf{x} \in \mathcal{X} \setminus C$, belongs to $\mathcal{H}_{at-most-k}$ and classifies all the points in C correctly. If $k < N_+ = 2k+1-N_-$ then $N_- \leq k$, and the classifier, defined as $h(\mathbf{x}_i) \coloneqq y_i$ for $i = 1, \ldots, 2k+1$ and $h(\mathbf{x}) = 1$ for $\mathbf{x} \in \mathcal{X} \setminus C$, belongs to $\mathcal{H}_{at-most-k}$ and classifies all the points in C correctly. This proves $\text{VCdim}(\mathcal{H}_{at-most-k}) \geq 2k+1$.

Suppose $2k+1 > |\mathcal{X}|$, we show that $\text{VCdim}(\mathcal{H}_{at-most-k}) \geq |X|$. Consider the set $|\mathcal{X}|$. Consider a labelling $\{y_1, \ldots, y_{|X|}\}$ with values in $\{0,1\}$. We have $N_+ \coloneqq \sum_{i=1}^{\ell} y_i$ elements of $|\mathcal{X}|$ with labels 1, and $N_- = |\mathcal{X}| - N_+$ elements with labels 0. Since $N_+ + N_- = |\mathcal{X}| < 2k+1$, we must either have $N_+ \leq k$ or $N_- \leq k$. If $N_+ \leq k$, then the classifier, defined as $h(\mathbf{x}_i) \coloneqq y_i$ for $i = 1, \ldots, |\mathcal{X}|$, belongs to $\mathcal{H}_{at-most-k}$ and classifies all the points in $\mathcal{X}$ correctly. If $N_- \leq k$, then the classifier, defined as $h(\mathbf{x}_i) \coloneqq y_i$ for $i = 1, \ldots, |\mathcal{X}|$, belongs to $\mathcal{H}_{at-most-k}$ and classifies all the points in $\mathcal{X}$ correctly. This proves $\text{VCdim}(\mathcal{H}_{at-most-k}) \geq |\mathcal{X}|$.

We have proved that $\text{VCdim}(\mathcal{H}_{at-most-k}) \geq \min\{2k+1, |\mathcal{X}|\}$.

We now prove that $\text{VCdim}(\mathcal{H}_{at-most-k}) < \min\{2k+2, |\mathcal{X}|+1\}$. Since $\mathcal{X}$ is finite, the fact that $\text{VCdim}(\mathcal{H}_{at-most-k}) \leq |\mathcal{X}|\}$ holds by definition of the VC-dimension. We now prove that $\text{VCdim}(\mathcal{H}_{at-most-k}) < 2k+2$. Consider a set of $2k+2$ elements, with $k+1$ elements labelled as 1 and $k+1$ elements labelled as 0. By definition of $\mathcal{H}_{at-most-k}$, there does not exist a classifier in $\mathcal{H}_{at-most-k}$ which classifies all those points correctly. We deduce that $\text{VCdim}(\mathcal{H}_{at-most-k}) < 2k+2$.

We conclude that $\text{VCdim}(\mathcal{H}_{at-most-k}) = \min\{2k+1, |\mathcal{X}|\}$.

Problem 5. (problem 6.7 in [SS14])

We have shown that for a finite hypothesis class $\mathcal{H}$, $\text{VCdim}(\mathcal{H}) \leq \lfloor \ln(|\mathcal{H}|) \rfloor$. *However, this is just an upper bound. The VC-dimension of a class can be much lower than that:*

1. *Find an example of a class $\mathcal{H}$ of functions over the real interval $\mathcal{X} = [0,1]$ such that $\mathcal{H}$ is infinite while* $\text{VCdim}(\mathcal{H}) = 1$.
2. *Give an example of a finite hypothesis class $\mathcal{H}$ over the domain $\mathcal{X} = [0,1]$, where* $\text{VCdim}(\mathcal{H}) = \lfloor \log_2(|\mathcal{H}|) \rfloor$.

1. Consider the class $\mathcal{H} \coloneqq \{h_y : y \in [0,1]\}$ consisting of indicator functions

$$h_y(x) = \mathbb{1}(x \leq y) = \begin{cases} 1 & \text{if } x \leq y, \\ 0 & \text{if } x > y, \end{cases} \tag{2}$$

for all $x \in [0,1]$.

Consider the single point $z = 0.5 \in [0,1]$. If z has label 1 then $h_1 \in \mathcal{H}$ classifies it correctly as $h_1(0.5) = \mathbb{1}(0.5 \leq 1) = 1$. If z has label 0 then $h_0 \in \mathcal{H}$ classifies it correctly as $h_0(0.5) = \mathbb{1}(0.5 \leq 0) = 0$. This proves that $\mathcal{H}$ shatters the set $\{0.5\}$ of dimension 1, hence $\text{VCdim}(\mathcal{H}) \geq 1$.

Consider a set consisting of two distinct elements $z_1, z_2 \in [0,1]$, where without loss of generality we assume $z_1 < z_2$. Consider the case where z_1 has label 0 and z_2 has label 1. The only functions in $\mathcal{H}$ which output 0 when evaluated at z_1 are $\{h_y : y < z_1\}$ since $h_y(z_1) = \mathbb{1}(z_1 \leq y)$, but for all those functions we have $h_y(z_2) = \mathbb{1}(z_2 \leq y) = 0$ as $y < z_1 < z_2$. We have proved that for any subset of $[0,1]$ of size 2 is not shattered by $\mathcal{H}$, that is, for any subset of $[0,1]$ of size

2 there exists a labelling such that every element of $\mathcal{H}$ fail to classify correctly the two points. This shows that $\mathrm{VCdim}(\mathcal{H}) < 2$. We deduce that $\mathrm{VCdim}(\mathcal{H}) = 1$ while $\mathcal{H}$ is infinite.

2. Consider the class $\mathcal{H} = \{h_{1/4}, h_{3/4}\}$ consisting of two indicator functions as defined in Eq. (2). Note that $\lfloor \log_2(|\mathcal{H}|) \rfloor = 1$.

Consider the single point $z = 0.5 \in [0, 1]$. If z has label 1 then $h_{3/4} \in \mathcal{H}$ classifies it correctly as $h_{3/4}(0.5) = \mathbb{1}(0.5 \leq 3/4) = 1$. If z has label 0 then $h_{1/4} \in \mathcal{H}$ classifies it correctly as $h_{1/4}(0.5) = \mathbb{1}(0.5 \leq 1/4) = 0$. This proves that $\mathcal{H}$ shatters the set $\{0.5\}$ of dimension 1, hence $\mathrm{VCdim}(\mathcal{H}) \geq 1$. The same proof as above shows that $\mathrm{VCdim}(\mathcal{H}) < 2$, we deduce that $\mathrm{VCdim}(\mathcal{H}) = 1 = \lfloor \log_2(|\mathcal{H}|) \rfloor$.

Problem 6.

On page 49 (Learning Theory slides) why did we formulate the VC-dimension of large margin classifiers with respect to a fixed input space $\mathcal{X}$ rather than for example $\{\mathbf{x} \in \ell_2 : \|\mathbf{x}\|_2 \leq R\}$?

We have

$$\mathcal{H}_{\mathcal{X},\Lambda} = \{\mathbf{x} \mapsto \mathrm{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) : \mathbf{w} \in \ell_2 \text{ satisfies } \|\mathbf{w}\|_2^2 \leq \Lambda \text{ and } |\langle \mathbf{w}, \mathbf{x} \rangle| \geq 1 \text{ for all } \mathbf{x} \in \mathcal{X}\}$$

If we consider $\mathcal{X} \coloneqq \{\mathbf{x} \in \ell_2 : \|\mathbf{x}\|_2 \leq R\}$, then there exists some $\tilde{\mathbf{x}} \in \mathcal{X}$ such that $\|\tilde{\mathbf{x}}\|_2 < \Lambda^{-1}$. For all $\mathbf{w} \in \ell^2$, by Cauchy-Schwarz inequality, we then have

$$|\langle \mathbf{w}, \mathbf{x} \rangle| \leq \|\mathbf{x}\|_2 \|\mathbf{w}\|_2 < \Lambda^{-1}\Lambda = 1.$$

Therefore, by definition of $\mathcal{H}_{\mathcal{X},\Lambda}$, for $\mathcal{X} \coloneqq \{\mathbf{x} \in \ell_2 : \|\mathbf{x}\|_2 \leq R\}$, the set $\mathcal{H}_{\mathcal{X},\Lambda}$ is empty. We reach the same conclusion when considering any set $\mathcal{X}$ for which there exists some $\tilde{\mathbf{x}} \in \mathcal{X}$ such that $\|\tilde{\mathbf{x}}\|_2 < \Lambda^{-1}$.

For this reason, on page 49 (Learning Theory slides), we formulate the VC-dimension of large margin classifiers with respect to a fixed input space $\mathcal{X}$ rather than for example $\{\mathbf{x} \in \ell_2 : \|\mathbf{x}\|_2 \leq R\}$.

Extra Problem (Littlestone dimension).

This problem is used to solve Problem 7, we present it separately as it is holds in a more general setting.

Suppose we wish to design a deterministic online algorithm with a mistake bound for learning with respect to some hypothesis class $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ where $\mathcal{Y} = \{-1, 1\}$. Suppose a deterministic algorithm $\mathcal{A}$ tries to learn a hypothesis (function) $h \in \mathcal{H}$ and is sequentially given the online training sequence s consisting of $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \mathcal{Y}$, then we denote the number of mistakes $\mathcal{A}$ makes by $M_{\mathcal{A}}(h, s)$. The maximum number of mistakes an algorithm $\mathcal{A}$ makes is $\max_{h \in \mathcal{H},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s)$ where $\mathcal{S}$ denotes the set of all possible online training sequences with elements in $\mathcal{X} \times \mathcal{Y}$. We claim that

$$\mathrm{VCdim}(\mathcal{H}) \leq \max_{h \in \mathcal{H},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s) \qquad \textit{for all possible deterministic online algorithm } \mathcal{A}.$$

Note that this is equivalent to $\mathrm{VCdim}(\mathcal{H}) \leq \mathrm{Ldim}(\mathcal{H})$ where the Littlestone dimension is defined as

$$\mathrm{Ldim}(\mathcal{H}) \coloneqq \min_{\mathcal{A}} \max_{h \in \mathcal{H},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s)$$

where the minimum is taken over all possible deterministic online algorithm $\mathcal{A}$.

Let $\mathrm{VCdim}(\mathcal{H}) = d \in \mathbb{N}$. We want to prove that

$$d \leq \max_{h \in \mathcal{H},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s) \qquad \text{for all possible deterministic online algorithm } \mathcal{A}$$

$$\forall \mathcal{A}\ \exists\, h \in \mathcal{H}\ \exists\, s \in \mathcal{S} : d \leq M_{\mathcal{A}}(h, s)$$

that is, that for all possible learning algorithms $\mathcal{A}$ there exist a hypothesis (function) $h \in \mathcal{H}$ and a training sequence $s \in \mathcal{S}$ such that $\mathcal{A}$ makes at least d mistakes when trying to learn h given the online training sequence s. Given any algorithm, we show how to pick such a hypothesis $h \in \mathcal{H}$ and such a sequence $s \in \mathcal{S}$. Since $d = \mathrm{VCdim}(\mathcal{H}) = \sup(|C| : \mathcal{H} \text{ shatters } C)$, there exists a set $C = \{x_1, \ldots, x_d\} \subseteq \mathcal{X}$ such that $\mathcal{H}$ shatters C. The algorithm tries to learn a hypothesis $h \in \mathcal{H}$ but has only access sequentially to $h(x_1), h(x_2), \ldots$ For $t = 1, \ldots, d$, we give the instance x_t to the algorithm, the algorithm predicts $\hat{y}_t \in \{-1, 1\}$, we select a new hypothesis $h \in \mathcal{H}$ such that $h(x_i) = y_i$ for $i = 1, \ldots, t-1$ and $h(x_t) = -\hat{y}_t$ (such an $h \in \mathcal{H}$ exists because $\mathcal{H}$ shatters C), and we give $y_t \coloneqq -\hat{y}_t$ to the algorithm. Changing $h \in \mathcal{H}$ in such a way is allowed because from the point of the view of the algorithm nothing changes as it has only access to $h(x_1), \ldots, h(x_{t-1})$ at time t and that these never change. By construction, the algorithm classifies x_i wrongly for $i = 1, \ldots, d$. Hence, the algorithm trying to learn the hypothesis $h \in \mathcal{H}$ (chosen at time $t = d$) given any training sequence $s \in \mathcal{S}$ starting with $(x_1, y_1), \ldots, (x_d, y_d)$ makes at least d mistakes. Hence, we have proved that

$$\mathrm{VCdim}(\mathcal{H}) \leq \max_{h \in \mathcal{H},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s) \qquad \text{for all possible deterministic online algorithm } \mathcal{A},$$

or equivalently, that $\mathrm{VCdim}(\mathcal{H}) \leq \mathrm{Ldim}(\mathcal{H})$. The VC-dimension is always upper bounded by the Littlestone dimension.

Problem 7.

Prove the two VC-dimension results on page 49 (Learning Theory slides). Given $\mathcal{X} \subset \ell_2$ and a $\Lambda \in (0, \infty)$, we define

$$\mathcal{H}_{\mathcal{X},\Lambda} = \{\mathbf{x} \mapsto \mathrm{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) : \mathbf{w} \in \ell_2 \text{ satisfies } \|\mathbf{w}\|_2 \leq \Lambda \text{ and } |\langle \mathbf{w}, \mathbf{x} \rangle| \geq 1 \text{ for all } \mathbf{x} \in \mathcal{X}\}$$

where $1/\|w\|_2$ is the margin.

1. *Prove that*
$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X},\Lambda}) \leq \Lambda^2 \max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|_2^2.$$
2. *Show that for every $\Lambda \in (0, \infty)$ there exists an $\mathcal{X}$ such that*
$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X},\Lambda}) = \left\lfloor \Lambda^2 \max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|_2^2 \right\rfloor.$$

1. We can apply the result proved in Problem Extra to obtain that

$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X},\Lambda}) \leq \max_{h \in \mathcal{H}_{\mathcal{X},\Lambda},\, s \in \mathcal{S}} M_{\mathcal{A}}(h, s) \qquad \text{for all possible deterministic online algorithm } \mathcal{A}.$$

In particular, this bound holds for the Perceptron Algorithm which is a deterministic online algorithm. So, the VC-dimension of $\mathcal{H}_{\mathcal{X},\Lambda}$ is upper-bounded by the maximum number of mistakes done by the Perceptron Algorithm, which we can itself upper-bound using the Novikoff Perceptron Bound (page 39 Online Learning slides) with margin $\gamma = 1/\|\mathbf{w}\|_2$. We obtain

$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X},\Lambda}) \leq \left(\frac{\max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|}{\gamma} \right)^2 = \|\mathbf{w}\|_2^2 \max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|^2 \leq \Lambda^2 \max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|_2^2.$$

2. Fix $\Lambda \in (0, \infty)$ and consider the set consisting of one element

$$\mathcal{X}_\Lambda \coloneqq \{\tilde{\mathbf{x}}\}, \qquad \tilde{\mathbf{x}} \coloneqq (\Lambda^{-1}, 0, 0, \dots) \in \ell^2.$$

We have $\|\tilde{\mathbf{x}}\|_2 = \Lambda^{-1}$. We then need to prove that

$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}) = \left\lfloor \Lambda^2 \max_{\mathbf{x}\in\mathcal{X}_\Lambda} \|\mathbf{x}\|_2^2 \right\rfloor = \left\lfloor \Lambda^2 \, \|\tilde{\mathbf{x}}\|_2^2 \right\rfloor = \left\lfloor \Lambda^2 \Lambda^{-2} \right\rfloor = 1.$$

Recall that

$$\begin{aligned}\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda} &= \{\mathbf{x} \mapsto \mathrm{sign}(\langle \mathbf{w}, \mathbf{x}\rangle) : \mathbf{w} \in \ell_2 \text{ satisfies } \|\mathbf{w}\|_2 \le \Lambda \text{ and } |\langle \mathbf{w}, \mathbf{x}\rangle| \ge 1 \text{ for all } \mathbf{x} \in \mathcal{X}_\Lambda\} \\ &= \{\tilde{\mathbf{x}} \mapsto \mathrm{sign}(\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle) : \mathbf{w} \in \ell_2 \text{ satisfies } \|\mathbf{w}\|_2 \le \Lambda \text{ and } |\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle| \ge 1\}.\end{aligned}$$

So, we have the condition $|\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle| \ge 1$. Using Cauchy-Schwarz inequality, we also have

$$|\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle| \le \|\mathbf{w}\|_2 \|\tilde{\mathbf{x}}\|_2 \le \Lambda\Lambda^{-1} = 1,$$

we deduce that we must have $|\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle| = 1$ which gives either $\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle = 1$ or $\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle = -1$. Those two cases arise when $\mathbf{w} \coloneqq \Lambda^2 \tilde{\mathbf{x}}$ and when $\mathbf{w} \coloneqq -\Lambda^2 \tilde{\mathbf{x}}$, respectively. Indeed, for both cases, we have $\|\mathbf{w}\|_2 = \Lambda^2 \|\tilde{\mathbf{x}}\|_2 = \Lambda^2 \Lambda^{-1} = \Lambda$ and $|\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle| = \Lambda^2 |\langle \tilde{\mathbf{x}}, \mathbf{x}\rangle| = \Lambda^2 \|\tilde{\mathbf{x}}\|_2^2 = \Lambda^2 \Lambda^{-2} = 1$.

Since the classifiers in $\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}$ are of the form $\tilde{\mathbf{x}} \mapsto \mathrm{sign}(\langle \mathbf{w}, \tilde{\mathbf{x}}\rangle)$, we deduce that $\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}$ consists of exactly two classifiers

$$\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda} = \big\{\tilde{\mathbf{x}} \mapsto 1, \tilde{\mathbf{x}} \mapsto -1\big\},$$

that is, the classifier which assigns 1 to $\tilde{\mathbf{x}}$ and the one which assigns -1 to $\tilde{\mathbf{x}}$. So, the element $\tilde{\mathbf{x}}$ can be correctly classified by a function $h \in \mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}$ for its two possible labellings. We deduce that $\mathrm{VCdim}(\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}) \ge 1$. Since $|\mathcal{X}_\Lambda| = 1$, we obtain that $\mathrm{VCdim}(\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}) = 1$. We have shown that

$$\mathrm{VCdim}(\mathcal{H}_{\mathcal{X}_\Lambda,\Lambda}) = 1 = \left\lfloor \Lambda^2 \max_{\mathbf{x}\in\mathcal{X}_\Lambda} \|\mathbf{x}\|_2^2 \right\rfloor.$$

Reference

[SS14] *Understanding Machine Learning from Theory to Algorithms*, S. Shalev-Shwartz and S. Ben-David, Cambridge University Press (2014), **link**.