EXAM 04/09/2019

EXERCISE N°1

POINT1

The binary classification problem is a supervised learning problem with :

- Domain set X which is the set of all possible objects to make predictions about, where a domain point $\vec{x} \in X$ is called instance and is usually represented by a vector of features
- Label set $Y = \{-1, +1\}$, that defines the set of all possible labels in this case the labels are only 2
- Model class H : is the set of functions that represents a mapping from X to Y
- loss function : is a function $l : HxZ \to R^+$ where $Z = XxY$ that, given an hypothesis provides a measure of how much we lose by predicting the value $h(\vec{x})$ for $\vec{x}$ instead of the correct value y
- Generalization error : $L_d(h) = E_{z\sim D}[l(h,z)]$ where D is the unknown probability distribution over Z from which $(x_i, y_i) \in S$ have been drawn (as independent samples).
- Training error : $L_S(h) = \frac{1}{m}\sum_{i=1}^{m} l(h,(x_i, y_i))$ where $S = \big((x_1, y_1) \ldots (x_m, y_m)\big)$ is the training set.

POINT2

An hypothesis class H is PAC learnable with respect to Z and a loss function $l : HxZ \to R^+$, if there exists a function $m_H: (0,1)^2 \to N$ (sample complexity) and a learning algorithm such that for every $\delta \in (0,1)$, $\varepsilon \in (0,1)$, for every distribution D over Z and for every true labeling function $f : X \to \{0,1\}$, if the realizability assumption holds with respect to H, D, f ; when running the learning algorithm on $m \geq m_H(\varepsilon, \delta)$ iid samples generated by D and labeled by f the algorithm returns an hypothesis h ,such that, with probability $\geq 1 - \delta :$ $L_{D,f}(h) \leq \varepsilon$.

POINT3

For instance, let H be an hypothesis class from domain X to $\{0,1\}$ and consider the 0-1 loss. Assume $VCdim(H) = d < +\infty$. Then there are two absolute constants C_1, C_2 such that :

$$C_1 \frac{\left(d + \log\left(\frac{1}{\delta}\right)\right)}{\varepsilon^2} \leq m_H(\varepsilon, \delta) \leq C_2 \frac{\left(d + \log\left(\frac{1}{\delta}\right)\right)}{\varepsilon^2}$$

EXERCISE N°2

POINT1

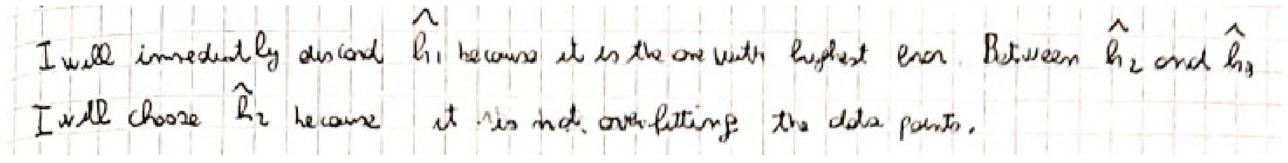
Given :

- Domain set X which is the set of all possible objects to make predictions about, where a domain point $x \in X$ is called instance and in this case is equal to R^d
- Label set $Y = R$

Given a training set $S = \big((\overrightarrow{x_1}, y_1) \ldots (\overrightarrow{x_m}, y_m)\big)$ with $\overrightarrow{x_i} \in X, y_i \in Y\ \forall i = 1,..,m$ we need to choose an hypothesis class H, which defines the possible models or classification rules, from which we can pick our model to make predictions, and a loss function $l : HxZ \to R^+$ where $Z = XxY$ that given an hypothesis provides a measure of how much we lose by predicting the value $h(\vec{x})$ for $\vec{x}$ instead of the correct value y. The goal is to find an hypothesis $\hat{h} \in H$ with low generalization error : $L_d(\hat{h}) = E_{z\sim D}[l(\hat{h}, z)]$ where :

- D is the unknown probability distribution over Z from which $(x_i, y_i) \in S$ have been drawn (as independent samples).
- $l(h, (\vec{x}, y) = (h(\vec{x}) - y)^2$ is the squared loss function

POINT2


I will immedietly discard $\hat{h}_1$ because it is the one with highest error. Between $\hat{h}_2$ and $\hat{h}_3$ I will choose $\hat{h}_2$ because it is not over-fitting the data points.

EXERCISE N°3

POINT1

SVM is a supervised learning method that is used to find a solution for the binary classification problem. Hard-SVM is the formulation of SVM that works when data are linearly separable, namely, with reference to binary classification, when $\forall\ i = 1, \dots, m\ y_i(< \vec{w}, \vec{x_i} > +b) > \ 0$ or equivalently when there exists an halfspace $(\vec{w}, \vec{x})$ such that $y_i = sign(< \vec{w}, \vec{x_i} > +b)\ \forall\ i = 1, \dots, m$ where $S = \left((\vec{x_1}, y_1) \dots (\vec{x_m}, y_m)\right)$ is the training set used to train the SVM.

The objective of the SVM is not only to find a separating hyperplane that perfectly classifies the data as done for other types of methods like perceptron, but it also finds the one that maximizes the margin i.e. the distance between the hyperplane and the closest sample to it in the training set.
Therefore the problem solved by Hard-SVM can be stated as follows :

Input : $S = \left((\vec{x_1}, y_1) \dots (\vec{x_m}, y_m)\right)$ linearly separable dataset

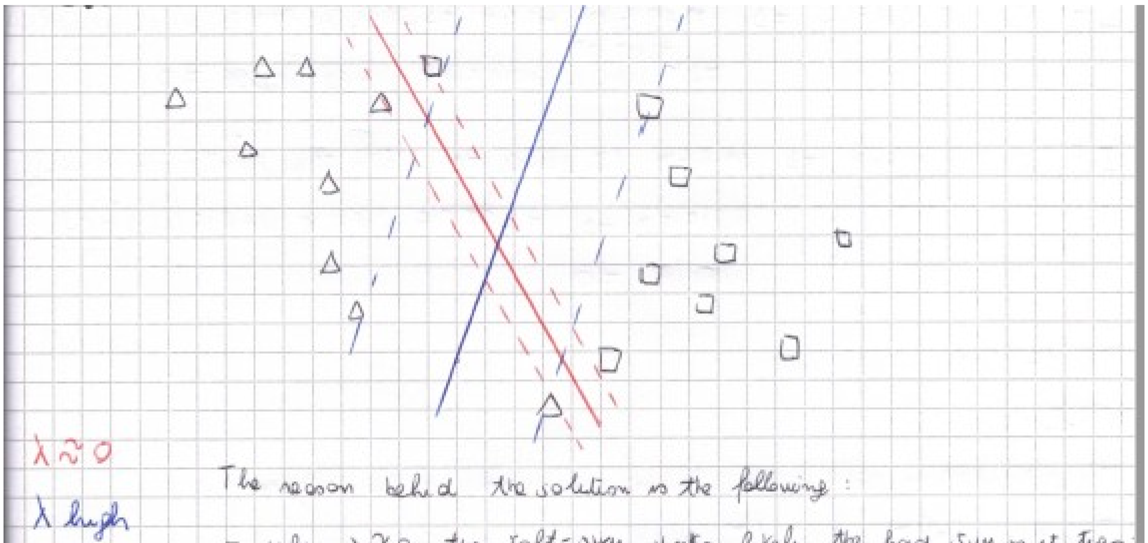
Goal : $argmax_{\substack{(\vec{w},b) \\ ||\vec{w}|| \\ i\in\{1,\dots,m\}}}\ \min|< \vec{w}, \vec{x_i} > +b|\ subject\ to\ \forall\ i = 1, \dots, m\ y_i(< \vec{w}, \vec{x_i} > +b) \geq 1$

Output : $\frac{\vec{w}}{||\vec{w}||}$; $\frac{b}{||\vec{w}||}$

POINT2

As we can see the main difference between the Soft-SVM formulation and Hard-SVM formulation can be found in the constraint and mainly in the objective function. In fact, the objective function of Soft-SVM is : $\lambda||\vec{w}||^2 + \frac{1}{m}\sum_{i=1}^{m}\xi_i$ where ξ_i are slack variables. As we can see, the soft-SVM formulation has two terms in the objective function instead of one. This thing is due to the fact that, hard-SVM is only able to treat linearly separable dataset while the soft formulation is also able to find a solution for non linearly separable one by allowing some violation in the constraints with the slack variables. Violations that can be described through the following constraints : $\forall\ i = 1, \dots, m\ y_i(< \vec{w}, \vec{x_i} > +b) \geq 1 - \xi_i\ and\ \xi_i \geq 0$.

POINT3



The reason behind the solution is the following:

- when $\lambda \approx 0$ the soft-SVM works likely the hard SVM so it tries
  classify correctly all the points without caring or caring very less ab
  the maximization of the margin

- When $\lambda >> 0$ then, the soft-SVM emphasizes more a bigger margin
than all the points correctly classified (i.e. training error).