TP 1

Prof. Stéphanie Allassonnière

TP 1: Reminder on Markov Chains – Stochastic gradient descent

Exercise 1 : Box-Muller and Marsaglia-Bray algorithm

Let R a random variable with Rayleigh distribution with parameter 1 and Θ with uniform distribution on $[0, 2\pi]$. We also assume that R and Θ are independent.

$$\forall r \in \mathbb{R}, \qquad f_R(r) = r \exp\left(-\frac{r^2}{2}\right) \mathbb{1}_{\mathbb{R}^+}(r)$$

1. Let X and Y such that

$$X = R\cos(\Theta)$$
 and $Y = R\sin(\Theta)$.

Prove that both X and Y have $\mathcal{N}(0,1)$ distribution and are independent.

2. Write an algorithm for sampling independent Gaussian distribution $\mathcal{N}(0,1)$.

Algorithm 1: Marsaglia-Bray algorithm

- 1 while $V_1^2 + V_2^2 > 1$ do
- | Sample U_1, U_2 independant r.v. with distribution $\mathcal{U}([0,1])$;
- Set $V_1 = 2U_1 1$ and $V_2 = 2U_2 1$.
- 4 end
- $\begin{array}{l} \mathbf{5} \ \, \mathrm{Set} \, \, S = \sqrt{-2 \log (V_1^2 + V_2^2)} \,\,; \\ \mathbf{6} \ \, \mathrm{Set} \, \, X = S \frac{V_1}{\sqrt{V_1^2 + V_2^2}} \,\, \mathrm{and} \,\, Y = S \frac{V_2}{\sqrt{V_1^2 + V_2^2}} \,\,; \end{array}$
- - **3.** Consider the algorithm given above.
 - a) What is the distribution of (V_1, V_2) at the end of the "while" loop?
 - **b**) What is the expected number of steps in the "while" loop?
 - c) Set

$$T_1 = \frac{V_1}{\sqrt{V_1^2 + V_2^2}},$$
 and $V = V_1^2 + V_2^2$.

Show that T_1 and V are independent, $V \sim \mathcal{U}([0,1])$ and T_1 has the same distribution as $cos(\Theta)$ with $\Theta \sim \mathcal{U}([0, 2\pi])$.

d) What is the distribution of the output (X, Y)?

Prof. Stéphanie Allassonnière

TP 1

Exercise 2: Invariant distribution

We define a Markov chain $(X_n)_{n\geq 0}$ with values in [0,1] as follows: given the current value $X_n \ (n \in \mathbb{N})$ of the chain,

• if $X_n = \frac{1}{k}$ (for some positive integer k), we let:

$$\begin{cases} X_{n+1} = \frac{1}{k+1} & \text{with probability } 1 - X_n^2 \\ X_{n+1} \sim \mathcal{U}([0,1]) & \text{with probability } X_n^2. \end{cases}$$

- if not, $X_{n+1} \sim \mathcal{U}([0,1])$.
- **1.** Prove that the transition kernel of the chain $(X_n)_{n\geq 0}$ is given by :

$$P(x,A) = \begin{cases} x^2 \int_{A \cap [0,1]} dt + (1-x^2) \delta_{\frac{1}{k+1}}(A) & \text{if } x = \frac{1}{k} \\ \int_{A \cap [0,1]} dt & \text{otherwise.} \end{cases}$$

where δ_{α} is the Dirac measure at α .

- **2.** Prove that the uniform distribution on [0,1] is invariant for P. In the following, this invariant distribution will be denoted by π .
- **3.** Let $x \notin \left\{\frac{1}{k}, \ker \in \mathbb{N}^*\right\}$. Compute the value of $Pf(x) = \mathbb{E}[f(X_1) \mid X_0 = x]$, for a bounded measurable function f. Deduce $P^n f(x)$ for all $n \geqslant 1$. Compute $\lim_{n \to +\infty} P^n f(x)$ in terms of $\int f(x)\pi(x) dx$.
- **4.** Let $x = \frac{1}{k}$ with $k \geqslant 2$.
 - a) Let $n \in \mathbb{N}^*$. Compute $P^n\left(x, \frac{1}{n+k}\right)$ in terms of k and n.
 - **b**) Do we have $\lim_{n \to +\infty} P^n(x, A) = \pi(A)$ when $A = \bigcup_{q \in \mathbb{N}} \left\{ \frac{1}{k+1+q} \right\}$?

Exercise 3: Stochastic Gradient Learning in Neural Networks, [Bot91, BCN16]

In the exercise, we consider the problem of classifying patterns x into two classes $y=\pm 1$. We assume that there is a relationship between a pattern and its class, embodied by some probability distribution $\mathbb{P}(x,y)$. If we know this distribution, we know the conditional probabilities $\mathbb{P}(y|x)$ as well, and we can solve immediately the problem using the Bayes decision rule. Learning means "Acquiring enough knowledge about $\mathbb{P}(x,y)$ from the examples to solve the classification problem".

The statistical machine learning approach begins with the collection of a sizeable set of examples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, where for each $i \in [1, n]$ the vector x_i represents the features and the scalar y_i a label indicating whether x_i belongs $(y_i = 1)$ or not $(y_i = -1)$ to a particular class. With such a set of examples, one can construct a classification program, defined by a prediction function h, and measure its performance by counting how often the program prediction $h(x_i)$ differs from the correct prediction y_i . To avoid rote memorization, one should aim to find a prediction function that generalizes the concepts that may be learned from the examples. One way to achieve good generalized performance is to choose amongst a carefully selected class of prediction functions.

Thanks to such a high-dimensional sparse representation of documents, it has been deemed empirically sufficient to consider prediction functions of the form $h(x; w, \tau) = {}^twx - \tau$. Here, twx is a linear discriminant parameterized by $w \in \mathbb{R}^d$ and $\tau \in \mathbb{R}$ is a bias that provides a way to compromise between precision and recall, $\mathbb{P}[y=1|h(x)=1]$ and $\mathbb{P}[h(x)=1|y=1]$ respectively. The accuracy of the predictions could be determined by counting the number of times that $sign(h(x; w, \tau))$ matches the correct label, i.e., 1 or -1. However, while such a prediction function may be appropriate for classifying new features, formulating an optimization problem around it to choose the parameters $(w; \tau)$ is impractical in large-scale settings due to the combinatorial structure introduced by the sign function, which is discontinuous. Instead, one typically employs a continuous approximation through a loss function that measures a cost for predicting h when the true label is y.

An **Adaline** (Widrow and Hoff, 1960) actually learns by (i) considering linear prediction function, $h(x, w) = {}^twx$, and (ii) measuring the quality of the system through the mean squared error:

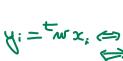
 $C_{Adaline}(w) = \int (y - h(x, w))^2 d\mathbb{P}(x, y) = \int (y - {}^twx)^2 d\mathbb{P}(x, y).$

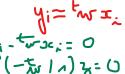
Learning consists of finding the parameter w^* that minimizes the above, or a more general, cost. This framework is the basis of classical statistical inference theory. Hundreds of practical algorithms have been derived.

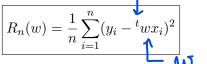
In the following, we will denote by z = (x, y) the observation and consider the cost or expected risk given a parameter vector w with respect to the probability \mathbb{P}

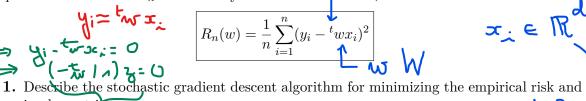
$$R(w) = \mathbb{E}[J(w,z)] = \int (y - {}^t wx)^2 d\mathbb{P}(z).$$

While it may be desirable to minimize the expected loss that would be incurred from any inputoutput pair, such a goal is untenable when one does not have complete information about P. Thus, in practice, one seeks the solution of a problem that involves an estimate of the expected risk R. In supervised learning, one has access (either all-at-once or incrementally) to a set of $n \in N$ independently drawn input-output samples $\{z_i = (x_i, y_i)\}_{i=1}^n$ and one may define the empirical risk function $R_n : \mathbb{R}^d \to \mathbb{R}$ by

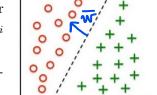








- implement d'inéaire d=2
- le noyau d'une forme linéaire est un hyperplan **2.** Sample a set of observations $\{z_i\}_{i=1}^n$ by generating a collection of random points x_i of \mathbb{R}^2 , $\bar{w} \in \mathbb{R}^2$ seen as the normal vector of an hyperplane, \times straight line here, and assigning the label y_i according to the side of the hyperplane the point x_i is.



- 3. Test the algorithm you wrote at the first question over these observations. What is the vector w^* estimated? Is it far from \bar{w} ?
- **4.** Noise your observations $\{z_i\}_{i=1}^n$ with an additive Gaussian noise and perform the optimisation again. Compare with the result of question three.
- 5. Test the algorithm on the Breast Cancer Wisconsin (Diagnostic) Data Set [WSM95]: http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29.

Références

- [BCN16] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for largescale machine learning. eprint arXiv:1606.04838, 2016.
- [Bot91] Léon Bottou. Stochastic gradient learning in neural networks. In Neuro-Nîmes 91, 1991.
- [WSM95] William H. Wolberg, W. Nick Street, and Olvi L. Mangasarian. UCI machine learning repository, 1995.