Entropic Variable Projection for Explainability and Intepretability

François Bachoc[1], Fabrice Gamboa[1], Max Halford[1,2], Jean-Michel Loubes[1], Laurent Risser[3]

[1] Institut de Mathématiques de Toulouse (UMR 5219), Université de Toulouse, F-31062 Toulouse, France
[2] Institut de Recherche en Informatique de Toulouse (UMR 5505), Université de Toulouse, F-31062 Toulouse, France
[3] Institut de Mathématiques de Toulouse (UMR 5219), CNRS, France

Abstract

In this paper, we present a new explainability formalism designed to explain how the possible values of each input variable in a whole test set impact the predictions given by black-box decision rules. This is particularly pertinent for instance to temper the trust in the predictions when specific variables are in a sensitive range of values, or more generally to explain the behaviour of machine learning decision rules in a context represented by the test set. Our main methodological contribution is to propose an information theory framework, based on entropic projections, in order to compute the influence of each input-output observation when emphasizing the impact of a variable. This formalism is thus the first unified and model agnostic framework enabling to interpret the dependence between the input variables, their impact on the prediction errors, and their influence on the output predictions. Importantly, it has in addition a low algorithmic complexity making it scalable to real-life large datasets. We illustrate our strategy by explaining complex decision rules learned using XGBoost and Random Forest classifiers. We finally make clear its differences with explainability strategies based on single observations, such as those of LIME or SHAP, when explaining the impact of different pixels on a deep learning classifier using the MNIST database.

1 Introduction

Machine learning algorithms build predictive models which are nowadays used for a large variety of tasks. Over the last decades, the complexity of such algorithms has grown, going from simple and interpretable prediction models based on regression rules to very complex models such as random forests, gradient boosting, and models using deep neural networks. We refer to Trevor et al. [1] for a description of these methods. Such models are designed to maximize the accuracy of their predictions at the expense of the interpretability of the decision rule. Little is also known about how

the information is processed in order to obtain a prediction, which explains why such models are widely considered as black boxes.

This lack of interpretability gives rise to several issues. When an empirical risk is minimized, the training procedure may be unstable or highly dependent on the optimization procedure due to *e.g.* convexity and unimodality. Another subtle, though critical, issue is that the optimal decision rules learned by a machine learning algorithm highly depend on the properties of the learning sample. If a learning sample presents unwanted trends or a bias, then the learned decision rules will reproduce these trends or bias, even if there is no intention of doing so. As a consequence, many users express a lack of trust in these algorithms. The European Parliament even adopted a law called GDPR (General Data Protection Regulation) to protect the citizens from decisions made without the possibility of explaining why they were taken, introducing a right for explanation in the civil code. Hence, building intelligible models is nowadays an important research direction in data science.

Different methods have been proposed to make understandable the reasons leading to a prediction, each author using a different notion of explainability and interpretability. We mention early works by Herlocker et al. [2] for recommender systems, Craven and Shavlik [3] for neural networks Dzindolet et al. [4], or Lou et al. [5] for generalized additive models. Another generic solution, described in Baehrens et al. [6] and Caruana et al. [7], focused on medical applications. Recently, a special attention has also been given to deep neural systems. We refer for instance to Montavon et al. [8], Selvaraju et al. [9] and references therein. Clues for real-world applications are given in Hall et al. [10]. Ribeiro et al. [11] (LIME) recently proposed to locally mimic a black-box model and then to give a feature importance analysis of the variables at the core of the prediction rule. In Lipton [12], a discussion was recently opened to refine the discourse on interpretability. In Koh and Liang [13] the authors finally proposed a strategy to understand black-box models, as we do, but in a parametric setting.

Our conception of the notion of interpretability for machine learning algorithms is the ability to quantify the specific influence of each of the $p \geq 1$ variables in a test set. We specifically determine the global effect of each variable in the learning rule and how a particular variation of this variable affects the accuracy of the prediction. This allows to understand how the predictions evolve when a characteristic of the observations is modified. To achieve this, we propose in this paper a sensitivity analysis strategy for machine learning algorithms. It is directly inspired by the field of sensitivity analysis for computer code experiments (see *e.g.* Lemaître et al. [14]), where the relative importance of the input variables involved in an abstract input-output relationship modeling a computer code is computed Saltelli et al. [15].

We emphasize that contrary to *e.g.* Ribeiro et al. [11] or Lundberg and Lee [16] (SHAP), our method deals with global explainability since it quantifies the global effect of the variables for all the test samples instead of individual observations. We also highlight that our point of view is different from previous works where the importance of each variable was also considered. Sparse models (see for instance [17] for a general introduction on the importance of sparsity) enable to identify important variables. Importance indicators have also been developed in machine learning to detect which variables play a key role in the algorithm. For instance, importance of variables is often computed using feature importance or Gini indices (see in [18] or [1]). Yet such indexes

are computed without investigating the particular effects of each variable and without explaining their particular role in the decision process. We also strongly believe that running the algorithm over observations which are created artificially by increasing stepwise the value of a particular variable is not desirable solution. By doing so, the correlations between variables are indeed not taken into account. Moreover, newly generated observations may be outliers with respect to the learning and test samples.

The paper falls into the following parts: Methodology is explained Sections 2 and 3. Results are given Section 4 and the discussions are finally developed Section 5.

2 Optimal perturbation of Machine Learning datasets

We first consider a test set $(X_i^1, \ldots, X_i^p, \hat{Y}_i, Y_i)$ for $i = 1, \ldots, n$, where $\hat{Y}_i = f(X_i^1, \ldots, X_i^p)$ is the prediction made by black box decision rules $f : \mathbb{R}^p \to \mathbb{R}$ on the input observation $X_i = (X_i^1, \ldots, X_i^p)$, and Y_i is the true output. Our goal is to quantify the impact of each input variable X^{j_0} with respect to different quantities of interest such as *e.g.* the error rate, the classification proportions, or the mean and variance in regression. To achieve this task, the key idea of this paper is to quickly compute a set of weights $\{\lambda_i^{j_0}\}_{i=1,\ldots,n}$ which will stress the distribution made of the triples $(X_i, \hat{Y}_i, Y_i)$ with respect to a property on the $\{X^{j_0}\}_{i=1,\ldots,n}$ (*e.g.* their mean). The quantity of interest will then be computed on the re-weighted test set. A fundamental aspect of our methodology is that we constrain the distribution of the re-weighted test set to be as close as possible to the empirical distribution of the original test set

$$Q_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{X_i, \hat{Y}_i, Y_i}\,, \tag{1}$$

making the problem well posed.

2.1 Optimal perturbation of distributions under moment constraints

In order to experience and to explore the behavior of a predictive model, a natural idea is to study its response to stressed inputs. In very particular, stressing the values of a particular variable while preserving the global distribution of the test data, enables to monitor the response of the algorithm to such entries and thus to explain the influence of this variable. Hence, we consider a probability distribution Q on an abstract measurable polish space $(E, \mathcal{B}(E))$, where Q represents here the distribution of the original test data. There exist different solutions to create modifications of a probability measure. In this paper, we consider an information theory framework in which we stress the mean value of a given variable while minimizing its Kullback-Leibler information (also called mutual entropy) with respect to the initial distribution Q. We then obtain a distribution Q_t, where t is a parameter controlling the amount of stress in the sense that $t = t_0$ means no perturbation.

First, let us recall the definition of the Kulback-Leibler information. Let $(E, \mathcal{B}(E))$ be a measurable space and Q a probability measure on E. If P is another probability measure on $(E, \mathcal{B}(E))$, then the Kullback-Leibler information $KL(P, Q)$ is defined as

equal to $\int_E \log \frac{\mathrm{d}P}{\mathrm{d}Q}\,\mathrm{d}P$, if $P \ll Q$ and $\log \frac{\mathrm{d}P}{\mathrm{d}Q} \in L^1(P)$, and equal to $+\infty$ otherwise. For a given $k \geq 1$, let $\Phi : E \to \mathbb{R}^k$ be a measurable function representing the shape of the stress deformation. We assume that Φ is Q-almost surely full rank. Let then $t_0 = \int_E \Phi(x)\,\mathrm{d}Q(x)$ (whenever it exists) be the parameter that represents no deformation. We set for two vectors of $x, y \in \mathbb{R}^k$ the scalar product as $\langle x, y\rangle = x^\top y$. Let $\mathbb{P}_{\Phi,t}$ be the set of all probability measures P on $(E, \mathcal{B}(E))$ such that $\int_E \Phi(x)\,\mathrm{d}P(x) = t$. The constraint $P \in \mathbb{P}_{\Phi,t}$ ensures that the mean of Φ under P deviates from the mean of the initial distribution Q by a quantity $t - t_0$. Let also

$$Q_t := \arg \inf_{P \in \mathbb{P}_{\Phi,t}} KL(P, Q) \tag{2}$$

whenever it exists.
The following theorem is a simplified version of the Theorems in [19] and in [20], which provides a close form for the solution of the previous optimization problem.

Theorem 2.1. *Assume that $\mathbb{P}_{\Phi,t}$ contains a probability measure that is mutually absolutely continuous with respect to Q. For a vector $\xi \in \mathbb{R}^k$, let $Z(\xi) := \int_E e^{\langle \xi, \Phi\rangle}\,\mathrm{d}Q(x)$. We assume that the set on which Z is finite is open. Define now $\xi(t)$ as the unique minimizer of the strictly convex function $H(\xi) := \log Z(\xi) - \langle \xi, t\rangle$. Then, Q_t defined as solution to* (2) *exists and is unique. Furthermore, it can be computed as*

$$Q_t = \frac{\exp\langle \xi(t), \Phi\rangle}{Z(\xi(t))} Q. \tag{3}$$

This theorem will be used to create new distributions on the test set. We consider the case where Q is equal to Q_n, as defined in Eq. (1), so $\Phi : \mathbb{R}^{p+2} \to \mathbb{R}^k$ and is assumed to be continuous.

The next theorem, which is a corollary of Theorem 2.1, gives the entropic projection of Q_n with a stress at level $t - t_0$. This probability measure corresponds to the stressed distribution.

Theorem 2.2. *Let Φ and $t \in \mathbb{R}^k$ be such that t can be written as a convex combination of $\Phi(X_1, \hat{Y}_1, Y_1), \ldots, \Phi(X_n, \hat{Y}_n, Y_n)$, with positive weights. Then $\xi(t)$ exists and is unique. For $i = 1, \ldots, n$, let $\lambda_i^{(t)}$ be defined as*

$$\lambda_i^{(t)} = \exp\Big(\langle \xi(t), \Phi(X_i, \hat{Y}_i, Y_i)\rangle - \log Z(\xi(t))\,.\Big) \tag{4}$$

Then, the distribution of $P_{\Phi,t}$ having the smallest KL divergence with respect to Q_n is supported by $(X_1, \hat{Y}_1, Y_1), \ldots, (X_n, \hat{Y}_n, Y_n)$, and may be written as

$$Q_t = \frac{1}{n} \sum_{i=1}^{n} \lambda_i^{(t)} \delta_{X_i, \hat{Y}_i, Y_i}.$$

Notice that the previous theorem follows directly from Theorem 2.1. Indeed, the constraint qualification is assumed (existence of positive weights), and the continuity of Φ warrants that Z is defined on the whole space (the support of Q is compact).

A particularly appealing aspect of this approach is that Q_t is supported by the same observations as Q_n (which was defined as equal to Q), the constraint only leading to different weights for the observations. As a consequence, sampling new stressed test sets does not require to create new input-output observations $(X_i, \hat{Y}_i, Y_i)$ but only to compute the weights $\lambda_i^{(t)}$. This can be solved very quickly using Eq. (4). This makes it possible to deal with very large databases without computing new values for new observations. This differs from existing techniques based on perturbed observations as *e.g.* in LIME [11], where the data used for testing are created by changing randomly the labels or by bootstrapping the observations. Hereafter, we show how to use entropic projection approximations which provides more flexibility and ensures that the new distributions are very similar to the distribution of the original data and yet undergoing the desired stress condition.

2.2 Application to Machine Learning

We now apply Theorem 2.2 to special deformations that will enable to highlight the behavior of the classifiers when specific variables are stressed. Although this theorem is flexible enough to model various types of stress, we stress in this paper the mean of a particular variable, meaning that Φ is valued in $\mathbb{R}$ (*i.e.* $k = 1$). We specifically stress each variable X^{j_0} using the model $t = m_{j_0} + \epsilon$, where $m_{j_0} = \frac{1}{n}\sum_{i=1,\dots,n} X_i^{j_0}$ and $\epsilon \in \mathbb{R}$. Note that the level of stress $\epsilon \in \mathbb{R}$ should obviously satisfy $\min_{i=1,\dots,n} X_i^{j_0} < m_{j_0} + \epsilon < \max_{i=1,\dots,n} X_i^{j_0}$ to be admissible. We then have here $Z(\xi) = \frac{1}{n}\sum_{j=1}^{n} \exp(\xi X_i^j)$ and $\lambda_i^{(t)} = \exp(\xi(m_{j_0} + \epsilon)X_i^j - Z(\xi))$, for $i = 1, \dots, n$.

In order to simply tune ϵ when the values of the different studied variables X^{j_0} are heterogeneously distributed, we propose to quantify the stress level in terms of quantiles: We consider the empirical quantile function q_{j_0} associated to variable X^{j_0}, so $q_{j_0}(\rho) = X^{j_0}_{\sigma([n\rho])}$ for $0 \le \rho < 1$ and $\sigma(.)$ a function ordering the sample, *i.e.* $X^{j_0}_{\sigma(0)} \le X^{j_0}_{\sigma(1)} \le \dots \le X^{j_0}_{\sigma(n-1)}$. Then, the range of the stress level will be in $[q_{j_0}(\alpha), q_{j_0}(1-\alpha)]$, where $\alpha \in (0, 1/2)$ (a typical value is 0.05). We then tune ϵ as equal to $\epsilon_{j_0,\tau}$, where $\epsilon_{j_0,\tau} = \tau(m_{j_0} - q_{j_0}(\alpha))$ if $\tau \in [-1, 0]$, and $\epsilon_{j_0,\tau} = \tau(q_{j_0}(1-\alpha) - m_{j_0})$ if $\tau \in [0, 1]$. Parameter τ therefore allows to intuitively parameterize the level of stress whatever the distribution of the $\{X_1^{j_0}, \dots, X_n^{j_0}\}$. More precisely, $\tau = 0$ yields no change of mean, $\tau = -1$ changes the mean from m_{j_0} to the (small) quantile $q_{j_0}(\alpha)$ and $\tau = 1$ changes the mean from m_{j_0} to the (large) quantile $q_{j_0}(1-\alpha)$. In what follows, we denote $\lambda_i^{(j_0,\tau)}$ a weight $\lambda_i^{(t)}$ computed using parameter τ on variable X^{j_0}.

Theorem 2.3. *$Q_{n,j_0,\tau}$ the solution of the minimization program $\min_\nu KL(\nu, Q_n)$ under the constraint $\mathbb{E}_\nu(X^{j_0}) = m_{j_0} + \epsilon_{j_0,\tau}$, exists and can be uniquely computed as $Q_{n,j_0,\tau} = \frac{1}{n}\sum_{i=1}^{n} \lambda_i^{(j_0,\tau)} \delta_{X_i,\hat{Y}_i,Y_i}$.*

The proof of this theorem comes from Theorem 2.2 and considering $\Phi(\nu) = X^{j_0}$ and $t = m_{j_0} + \epsilon_{j_0,\tau}$. This theorem enables to re-weight the observations of each variable so that its mean increases or decreases. This is then used Section 3 to understand the particular role played by each variable. Note that the formulation of the reweighing problem 2.2 is very general and enables to consider a wide class of deformations. For

instance, let us consider an optimal re-weighting of the data, such that the mean of the variable i is changed to $m_i \in \mathbb{R}$, the mean of the variable j is changed to $m_j \in \mathbb{R}$ and the covariance between the variables i and j is changed to $c_{i,j}$. Then, the solution according to Eq. (4) and Theorem 2.2 is to let $\Phi(X^1, \ldots, X^p, \hat{Y}, Y) = (X^i, X^j, {}^i X^j)^\top$ and $t = (m_i, m_j, c_{i,j} + m_i m_j)^\top$.

3 Explainable models using perturbed distributional entries

In this section we consider that the transformation Φ and the level of stress $\varphi(\epsilon)$ indexed by a value τ have been chose as in Theorem 2.3. Denote by $\lambda_i^{(j_0,\tau)}$ the weights corresponding to a stress of intensity τ for the variable j_0 of a test set $(X_i, \hat{Y}_i, Y_i)_{i=1,\ldots,n}$. The first step consisting in computing these weights have been done. In the following, we detail the use of such technics to explain two classic situations encountered in machine learning: binary classification and multi-class classification. The regression case is also explained in Appendix A.

3.1 The case of binary classification

Consider the case where the Y_i and $\hat{Y}_i = f(X_i)$ belong to $\{0, 1\}$ for all $i = 1 \ldots, n$. This case corresponds to the binary classification problem for which the usual loss function is $\ell(Y, f(X)) = \mathbf{1}\{Y \neq f(X)\}$. We suggest to use the indicators described hereafter for the perturbed distributions $\frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \delta_{(X_i, \hat{Y}_i, Y_i)}$. Explaining the decision rules consists in quantifying the evolution of the error rate on each variable j_0 w.r.t. τ, hence the first indicator is the error rate, *i.e.*

$$\mathrm{ER}_{j_0,\tau} = \frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \mathbf{1}\{f_n(X_i) \neq Y_i\}.$$

We naturally suggest to plot $\mathrm{ER}_{j_0,\tau}$ as a function of τ for $\tau \in [-1, 1]$ and for each $j_0 \in \{1, \ldots, p\}$, as shown Fig. 1-(bottom). Remark that $\tau = 0$ corresponds to the algorithm performance baseline, without any perturbation of the test sample. In terms of interpretation, plotting $\mathrm{ER}_{j_0,\tau}$ highlights the variables which produce the largest amount of confusion in the error, *i.e.* those for which the variability among the two predicted classes is the most important, hampering the prediction error rate. The False and True Positive Rates may alternatively be represented using

$$\mathrm{FPR}_{j_0,\tau} = \frac{\frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \mathbf{1}\{Y_i \neq 1\}}{\frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \mathbf{1}\{f_n(X_i)=1\}} \quad \text{and} \quad \mathrm{TPR}_{j_0,\tau} = \frac{\frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \mathbf{1}\{f_n(X_i)=1\}}{\frac{1}{n}\sum_{i=1}^n \lambda_i^{(j_0,\tau)} \mathbf{1}\{Y_i=1\}}.$$

A ROC curve corresponding to perturbations of the variable j_0 can then be obtained by plotting pairs $(\mathrm{FPR}_{j_0,\tau}, \mathrm{TPR}_{j_0,\tau})$ for a large number of $\tau \in [-1, 1]$. We then obtain the evolution of both errors when τ evolves, which allows a sharper analysis of the error evolution (see *e.g.* Appendix B.4). Finally, the variables influence on the predictions

may be quantified by computing the proportion of predicted 1s

$$\mathrm{P1}_{j_0,\tau} = \frac{1}{n}\sum_{i=1}^{n} \lambda_i^{(j_0,\tau)} f_n(X_i)$$

which we suggest to plot similarly as $\mathrm{ER}_{j_0,\tau}$ (see Fig. 1-(top)). The figures representing these indicators make it possible to simply understand the particular influence of the variables to obtain a decision $Y = 1$, whatever the veracity of the prediction. Importantly, they point out which variables should be positively or negatively modified in order to change a given decision.

3.2 The case of multi-class classification

We now consider the case of a classification into k different categories, *i.e.* where Y_i and $f_n(X_i)$ belong to $\{1, \dots, k\}$ for all $i = 1, \dots, n$, and where $k \in \mathbb{N}$ is fixed. In this case, the strategy described for the binary classification can naturally be extended using

$$\mathrm{Pj}_{j_0,\tau} = \frac{1}{n}\sum_{i=1}^{n} \lambda_i^{(j_0,\tau)} \mathbf{1}\{f_n(X_i) = j\},$$

which denotes the portion of individuals assigned to the class j.

4 Results

In this section, we illustrate on the *Adult income* dataset[1] the use of the indices obtained using the entropic projection of samples on two classification cases: In subsection 4.1, X represents $n = 32000$ observations of dimension $p = 14$ and Y has 2 classes. Results of subsection 4.2 are obtained on the *MNIST* dataset[2], where X represents $n = 60000$ images of $p = 784$ pixels and Y has 10 classes. Note that the method accuracy is also assessed on synthetic data in Appendix B.2 and that the effect of 4 variables on the classification of the well known Iris dataset is shown in Appendix B.3. Importantly, the Python code to reproduce these experiments is freely available on GitHub[3].

4.1 Two class classification

In order to illustrate the performance of our procedure, we first consider the *Adult Income* dataset. It is made of about $n = 32000$ observations represented by $p = 14$ attributes (6 numeric and 9 categorical), each of them describing an adult. We specifically interpret the influence of 5 numeric variables on the categorical variable representing whether each adult has an income higher ($Y_i = 1$) or lower ($Y_i = 0$) than 50000$ per year.

We first trained three different classifiers (Logit Regression, XGboost and Random Forest[4]) on 25600 randomly chosen observations (80% of the whole dataset). We then

[1] https://archive.ics.uci.edu/ml/datasets/adult

[2] http://yann.lecun.com/exdb/mnist/

[3] [Subject to peer reviewed paper acceptance, private repository for now]

[4] *R* command *glm* and packages *xgboost* and *ranger*.

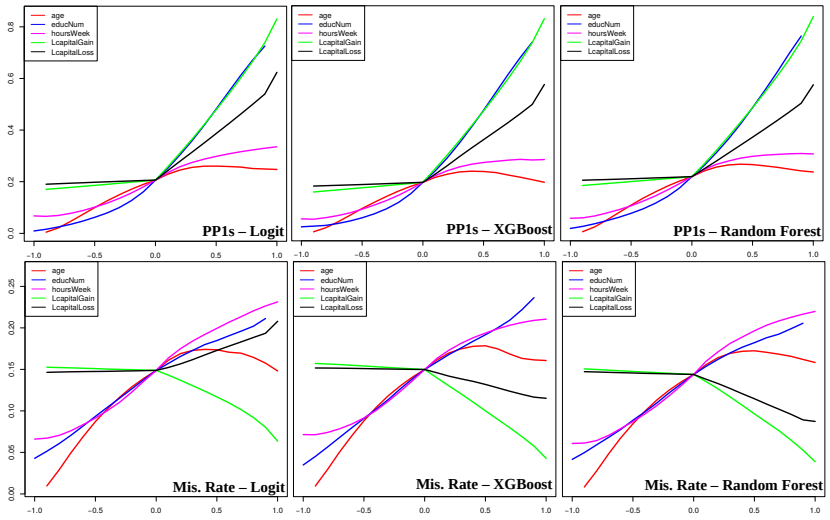

Figure 1: Results of Section 4.1 on the *Adult income* dataset. **(Top - PP1s)** Portion of predicted ones (*i.e.* High Incomes) with respect to the explanatory variable perturbation τ. **(Bottom - Mis. Rate)** Classification errors with respect to τ. There is no perturbation if $\tau = 0$. The larger (resp. the lower) τ, the larger (resp. the lower) the values of the selected explanatory variable.

performed the proposed sensitivity analysis strategy for each learned classifier on a test set made of the 6400 remaining observations. Detailed results are shown Fig. 1. Instead of only quantifying a score for each variable, we display in this figure the evolution of the algorithm confronted at gradually lower or higher values of τ for each variable. The curves were computed using 21 regularly sampled values of τ between -1 and 1 with $\alpha = 0.05$. For each variable, the weighted observations were therefore stressed so that their mean is contained between the 0.05 and 0.95 quantiles of the original (non-weighted) values distribution in the test set. Note that for a quick and quantified overview of the variables response to a positive (resp. negative) stress, the user can simply interpret the difference of the response for $\tau = 1.$ and $\tau = 0$ (resp. $\tau = 0$ and $\tau = -1.$), as illustrated Section 4.2 in the image case.

Influence of the variables in the decision rule We present in Fig. 1 *(Top)* the role played by each variable in the portion of predicted ones (*i.e.* high incomes) for the *Logit Regression*, *XGboost* and *Random Forest* classifiers. The curves Fig. 1 *(Top)* highlight the role played by the variable *educNum*. The more educated the adult, the higher his/her income will be. The two variables *LcapitalGain* and *LcapitalLoss* are also testimonial of high incomes since the adults having large incomes can obviously have more money than others on their bank accounts, or may easily contract debts, although the contrary is not true. It is worth pointing out the role played by the age variable which appears clearly in the figure: young adults earn increasingly large incomes with time, which is

well captured by the decision rules (left part of the red curves). For τ higher than about 0.25 (corresponding to 34 years old), being increasingly old is however not captured by the three tested decision rules to be related to higher incomes.

We emphasize that these curves enable to intuitively interpret the complex trends captured by *black-box* decision rules. They indeed quantify non-linear effects of the variables and very different behaviors depending on whether the variables increase or decrease. In other strategies designed to explain black-box decision rules, explainability is obtained by using the global feature importance indexes included in the decision rules. The *Feature Weights* count the number of times a feature appears in a classifier obtained by combining several classifiers (*e.g.* an ensemble of trees). For tree based methods, the *Gain* counts the average gain of splits using the feature, while the *Coverage* is based on the average coverage (number of samples affected) of splits using the feature. Implemented algorithms in Python or R enable to view these features importance, but they often contradict themselves as already quoted by several authors (see for instance in LIME [11]). Contrary to algorithms that study the influence of a variable by computing information theory criterion between different outputs of the algorithm for changes in the variables (see in Skater [21]), the variable changes we use are plausible since the stressed variables have distribution that are as close as possible to the initial distribution of true variables. Finally, we work directly on the real black-box model and do not approximate it by any surrogate model, as in [11].

Influence of the variables in the accuracy of the classifier Besides the influence of the variables on the algorithm outcome, it is worth studying their influence on the accuracy or veracity of the model. We then present in Fig. 1 *(Bottom)* the evolution of the classification error when each variable is stressed by τ. The three sub-figures (one for each prediction model) represents the evolution of the error confronted to the same modification of each variables. The error of the method on the original data is obtained for $\tau = 0$. Such curves point out which variables are the most sensitive to increasing prediction flaws. Such result may be used to temper the trust in the forecast depending on the values of the variables.

As previously, the curves appear as more informative than single scores: The three models enable to select the same couple of variables that are important for the accuracy of the prediction when they increase *i.e.* education number and numbers of hours worked pro week. The latter makes the prediction task the most difficult when it is increased. Indeed, the people working a large number of hours per week may not always increase their income, since it relies on different factors. People with high income however use to work a large number of weekly hours. Hence, these two variables play an important role in the prediction and their changes impact the prediction error. In the same flavor, more insight on the error terms could be done by dealing with the evolution of the False Positive Rate and True Positive Rate as presented in the Supplementary materials, Section B.4.

4.2 Image classification

We now measure the influence of pixel intensities in image recognition tasks. Each pixel intensity is treated as a variable and the stress is used to saturate the intensities towards one side of their spectrum (black if $\tau = 1$ or white if $\tau = -1$). We specifically trained a CNN on the MNIST dataset using a typical architecture that can be found on the Keras documentation[5]. The CNN was trained on a set of 60,000 images whilst the predictions were made on another 10,000 images. It achieved a test set accuracy north of 99%. For each of the 784 pixels j_0, we computed the $\{\lambda_i^{(j_0,\tau)}\}_{i=1,\ldots,10000}$ in the cases where τ equals -1 as well as 1, using the method of Section 2. The prediction proportions of each of the 10 digits was then computed using the method of Section 3.2. The whole process took around 9 seconds to run on a modern laptop (Intel Core i7-8550U CPU @ 1.80GHz, 24GB RAM) running Linux. The results are presented in Fig. 2-(top).

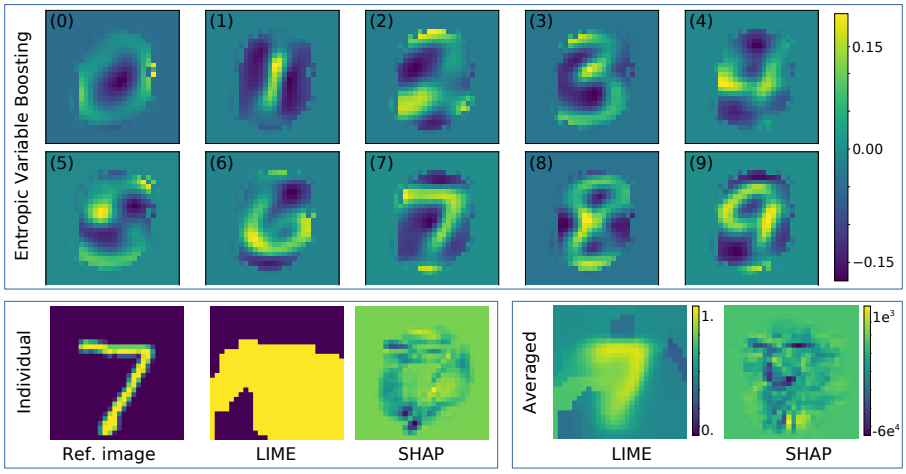

Figure 2: **(top)** Pixel contributions towards each digit according to our Entropic Variable Boosting method. **(bottom-left)** Pixel contributions to predict seven in an individual image representing a seven, using the LIME and SHAP packages. **(bottom-right)** Average pixel contributions to predict seven in all images of the MNIST test set representing a seven, using the LIME and SHAP packages.

The color of each pixel in Fig. 2-(top) represents it's contribution towards the prediction of each digit. For example, a value of 0.15 means that the CNN outputs, on average, a 0.15 higher probability when the associated pixel is activated. Although our method is pixel-based, it is still able to uncover regions which the CNN uses to predict each digit. Likewise, darker regions contain pixels that negatively correlated with each digit. Note that the edges of each image don't change color because the corresponding pixels have no impact whatsoever on the predictions. The left part of number 5 has pixels in common with number 6. However, we are able to see that the CNN identifies 6s

[5]https://keras.io/examples/mnist_cnn

by using the bottom part of the 6, more so than the top stroke which it uses to recognize 5s. Likewise, according to the CNN, the most distinguishing part of number 9 is the part that links the top ring with the bottom stroke.

We finally emphasize the main difference between our strategy and the two popular interpretability solutions LIME (Ribeiro et al. [11])[6] and SHAP (Lundberg and Lee [16])[7]: we work on whole test sets while these solutions interpret the variables (here pixels) influence when predicting specific labels in *individual observations*. As an illustration, Fig. 2-(bottom-left) represents the most influential pixels found with LIME and SHAP to predict a seven in an image of the MNIST test set representing a seven. To draw similar interpretations as those made on Fig. 2-(top), one can represent the average results obtained by using LIME or SHAP over all images of the MNIST test set representing a seven, as represented Fig. 2-(bottom-left). Note that the computations required about 7 and 10 hours using LIME and SHAP, respectively, which is much longer than when using our method (10 seconds). Averaged results are also less resolved for LIME and harder to interpret for SHAP. Our method can also natively compute other properties of the black-box decision rules with a negligible additional computational cost, as described in Section 3.

5 Conclusion

Explainability of *black-box* decision rules in the machine learning paradigm has many interpretations and has been tackled in a large variety of contributions. Here, we focused on the analysis of the variables importance and sensitivity and their impact on a decision rule. This work is inspired by computer code experiments. Actually, when building a surface response in computer code experiments, the prediction algorithm is applied to new entries to explore its possible outcomes. In the machine learning framework, the framework is quite different since the test input variables must follow the distribution of the learning sample. Therefore, evaluating the decision rule at all possible points does not make any sense. To coper with this issue, we have proposed an information theory procedure to stress the original variables without losing the information conveyed by the initial distribution. We proved that this solution amounts in re-weighting the observations of the test sample, leading to very fast computations and the construction of new indices to make clear the role played by each variable.

The first key advantage of this strategy is first to preserve as much as possible the distribution of the test set $(X_i^1, \ldots, X_i^p, \hat{Y}_i, Y_i)$, $i = 1, \ldots, n$ and thus preserving the correlations of the input variables that have then an impact on the indicators computed by using our procedure. In contrasts with other interpretability paradigms such as the PAC learning framework Valiant [22], we do not create artificial outliers. Its second key advantage is that, for a given perturbation, the weights are obtained by minimizing a one-dimensional convex function, for which the evaluation cost is $O(n)$. The total cost is then $O(np)$ for studying the impact of each of the p variables (see Appendix B.1) and there is no need to generate new data, nor even to compute new predictions from the black box algorithm, which is particularly costly if n or p is large. Our procedure

[6] https://github.com/marcotcr/lime
[7] https://github.com/slundberg/shap

therefore scales particularly well to large datasets as *e.g.* real-world image databases. Finally, the flexibility of the entropic variable projection procedure enables to study the response to various types of stress on the input variables (not only their mean but also their variability, joint correlations, ...) and thus to interpret the decision rules encountered in a wide range applications encountered in the field of Machine Learning.

References

[1] Hastie Trevor, Tibshirani Robert, and Friedman JH. The elements of statistical learning: data mining, inference, and prediction, 2009.

[2] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250. ACM, 2000.

[3] Mark Craven and Jude W. Shavlik. Extracting tree-structured representations of trained networks. In *Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, November 27-30, 1995*, pages 24–30, 1995.

[4] Mary T Dzindolet, Scott A Peterson, Regina A Pomranky, Linda G Pierce, and Hall P Beck. The role of trust in automation reliance. *International journal of human-computer studies*, 58(6):697–718, 2003.

[5] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 150–158, 2012. ISBN 978-1-4503-1462-6.

[6] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831, 2010.

[7] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1721–1730, 2015.

[8] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 2017.

[9] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? *arXiv preprint arXiv:1611.07450*, 2016.

[10] Patrick Hall, Navdeep Gill, and Mark Chan. Practical techniques for interpreting machine learning models: Introductory open source examples using python, h2o, and xgboost, 2018.

[11] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[12] Zachary C. Lipton. The mythos of model interpretability. In *Proceedings of the ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, pages 96–100, 2016.

[13] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, 2017.

[14] Paul Lemaître, Ekatarina Sergienko, Aurélie Arnaud, Nicolas Bousquet, Fabrice Gamboa, and Bertrand Iooss. Density modification-based reliability sensitivity analysis. *Journal of Statistical Computation and Simulation*, 85(6):1200–1223, 2015.

[15] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global sensitivity analysis: the primer*. John Wiley & Sons, 2008.

[16] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4765–4774, 2017.

[17] Peter Bühlmann and Sara Van De Geer. Introduction. In *Statistics for High-Dimensional Data*, pages 1–6. Springer, 2011.

[18] Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1):77–93, 2004.

[19] Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, pages 146–158, 1975.

[20] Imre Csiszár. Sanov property, generalized I-projection and a conditional limit theorem. *The Annals of Probability*, pages 768–793, 1984.

[21] Aaron Kramer and Pramit Choudhary. Model Interpretation with Skater. `https://datascienceinc.github.io/Skater/index.html`, 2018. [Online; accessed 10-Jan-2019].

[22] Leslie Valiant. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books (AZ), 2013.

[23] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Appendix

A Extension to the regression case

A.1 Methodology

As an extension to Section 3, we consider now the case of a real valued regression where $Y_i, f_n(X_i) \in \mathbb{R}$ for $i = 1 \ldots, N$. In order to understand the effects of each variable, first we consider, the mean criterion

$$\mathrm{M}_{i_0,\tau} = \frac{1}{N} \sum_{i=1}^{N} \lambda_i^{(i_0,\tau)} f_n(X_i),$$

which will indicate how a change in the variable will modify the output of the learned regression. Second the variance criterion

$$\mathrm{V}_{i_0,\tau} = \frac{1}{N} \sum_{i=1}^{N} \lambda_i^{(i_0,\tau)} \left(f_n(X_i) - \mathrm{M}_{i_0,\tau}\right)^2$$

is meant to study the stability of the regression with respect to the perturbation of the variables. Finally the root mean square error (RMSE) criterion

$$\mathrm{RMSE}_{i_0,\tau} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \lambda_i^{(i_0,\tau)} \left(f_n(X_i) - Y_i\right)^2}$$

is analogous to the classification error criterion since it enables to detect possibly misleading variable or confusing variables when learning the regression.
For each $i_0 \in \{1, \ldots, p\}$, these three criteria can be plotted as a function of τ for $\tau \in [-1, 1]$.

A.2 Application

We use now our strategy on the Boston Housing dataset[8]. This dataset deals with houses prices in Boston. It contains 506 observations with 13 variables that can be used to predict the price of the house to be sold. When considering an optimized Random Forest algorithm, the importance calculated as described in [23], enables to select the 5 most important variables as follows: *lstat* (15227), *rm* (14852), *dis* (2413), *crim* (2144) and *nox* (2042). Remark that the coefficients obtained using a linear model would lead to similar interpretations, with the the 5 most important variables as follows: *lstat* (-3.74), *dis* (-3.10), *rm* (2.67), *rad* (2.66), *tax* (-2.07)

As shown Fig. 3, our analysis goes further than this score. In particular we point out the non linear influence of the variables depending whether they are high or low. For instance the average number of rooms in a house (variable *rm*) is an important factor that makes the price increase in the case of large houses ($\tau > 0.$ in Fig. 3 *(Mean)*).

[8] https://www.kaggle.com/c/boston-housing

$Mean_0 - Mean_{-0.5}$
black (4.1)
rm (3.0)
dis (1.7)
zn (0.85)
...
age (-2.78)
indus (-3.2)
ptratio (-3.8)
lstat (-5.1)

$Mean_{0.5} - Mean_0$
rm (6.80)
zn (4.60)
chas (2.74)
dis (1.64)
...
rad (-2.99)
indus (-3.05)
tax (-3.18)
lstat (-5.26)

Table 1: Most responsive variables to a positive or negative stress τ when estimating House prices. Scores are shown between brackets and computed as the difference of the *Mean* curves of Fig. 3 for **(left)** $\tau = -0.5$ and $\tau = 0$, and **(right)** $\tau = 0$ and $\tau = 0.5$.

Interestingly, this is far less the case for smaller houses ($\tau < 0$. in Fig. 3 *(Mean)*) since there are other arguments than the number of rooms to keep a high price in this case. Although our methodology can highlight such complex learned properties, it can also find more simple properties such as the age for instance which has an almost linear influence for all values of τ .

Note that when the number of variables is large, the presence of too many curves may make the graph difficult to understand. In this cases, scores that represent average individual evolutions on given ranges of τ values for each variables can be computed. Then the highest and lowest scores can be represented as the most influential variables on the predictions. For instance, we represent Table. 1 the evolution of the Mean curves in Fig. 3 between $\tau = -0.5$ and $\tau = 0$, as well as between $\tau = 0$ and $\tau = 0.5$, which makes clearly understandable which are the most influential variables. It is important to remark that our methodology still allows that the learned decision rules won't be mainly influenced by the same variables depending on whether it the variable increases ($\tau > 0$) or decreases ($\tau < 0$). In Table. 1, the more influential variables are indeed *rm*, *lstat* and *zn* in the positive direction, while in the negative direction, the variables are *lstat*, *black* and *pratio*. Note that such variables are also cited in studies that relies on LIME [11] or SHAP [16] packages, but the curves we present are more informative and relies on the same distributional input.

B Additional results in the Classification case

B.1 Evaluation of the computational burden

We explained Section 5 that our strategy only optimizes, for each of the p variables, a function which evaluation cost is $O(n)$ with no additional outputs predictions out of the *black box* machine learning algorithm. To quantify this, we show Table 2 the computational times dedicated to the analysis of synthetic datasets having a different

amount of variables p and observations n. The variables interpretation was made using 21 values of τ, leading to curves as *e.g.* in Figure 1. Computations were run with Python on a standard Intel Core i7 laptop with 24GB memory and no parallelization. It appears that our strategy indeed has a $O(np)$ cost, so we then believe it may have a high impact to study the rules learned by black-box machine learning algorithms on large real-life datasets. Remark that when interpreting the influence of the pixel intensities on image test sets, as in Figure 2, only 3 values of τ are used. The computations are therefore about 7 times faster. This coherent with the 10 seconds required on 10000 MNIST images of 28×28 pixels in Section 4.2. Note finally that a preliminary implementation of our method in R has lead to very similar results.

p	n	time (sec)
10	10000	0.76
100	10000	7.79
1000	10000	82.5
10	100000	7.93
10	1000000	86.0

Table 2: Computational times required on synthetic datasets, where 21 levels of stress (τ) were computed on each of the p variables.

B.2 Results on simulated data

In order to further show that our procedure is able to properly recover the characteristics of machine learning algorithms, we agin tested it on synthetic data. We have run an experiment with $p = 5$ variables and $n = 10^6$ observations, where synthetic data are generated using a logistic regression model, with independent regressors and coefficient vector equal to $(-4, 2, 0, 2, 4)$. Fig. 4 clearly shows that our method enables to recover the signs and the hierarchy of the coefficients.

B.3 Results on the Iris dataset

As an additional assesment of the method on very well known and simple data, we now consider the *Iris* dataset[9]. This dataset is composed of 150 observations with 4 variables used to predict a label into three categories: *setosa*, *versicolor*, *virginica*. To predict the labels, we used an *Extreme Gradient Boosting* model and a *Random Forest* classifier. Results are show in Fig. 5. We first present for both models the Classification error. Then the two other subfigures show the effects of increasing or decreasing the 4 parameters, i.e the width or the length of the sepal or petal is shown for all classes. As expected, we recover the well known result that the width of the sepal is the main parameter which enables to differentiate the class *Setosa* while the differentiation between the two other remaining classes is less obvious.

[9] https://archive.ics.uci.edu/ml/datasets/iris

B.4 Other indices: ROC Curves

In the case of two class classification on the Adult Income dataset (Section 4.1), we have shown the evolution of the classification error when the stress parameter τ increases. Such results can straightforwardly be extended to True and false positive rates, which are commonly represented in ROC curves. Each point of these curves corresponds to the False Positive Rate and the True Positive Rate, for a sample drawn for each τ and each variable. All curves cross at the same point which corresponds to $\tau = 0$. It therefore becomes possible to study the evolution of each criterion.

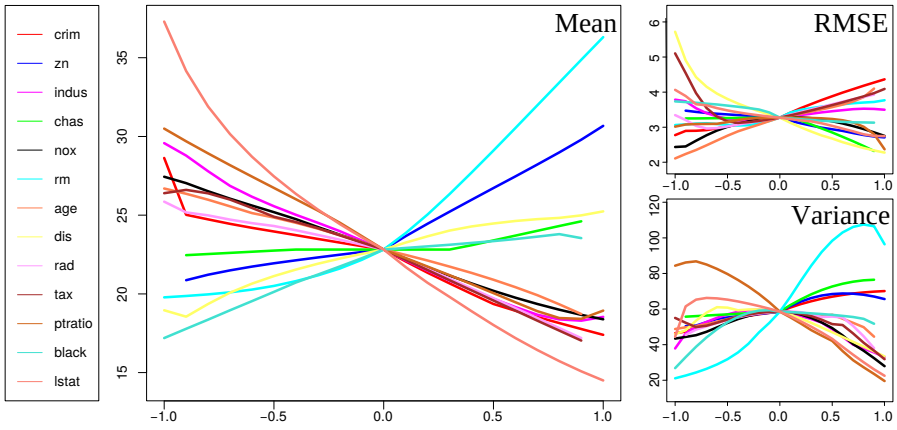

Figure 3: Results obtained on the *Boston Housing* dataset with Random Forests. The explanatory variable perturbation τ has the same signification as in Fig. 1.

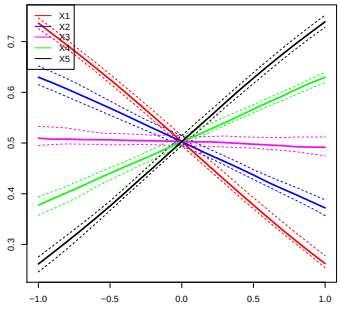

Figure 4: Proportion of ones found on synthetic data generated using a logistic regression model

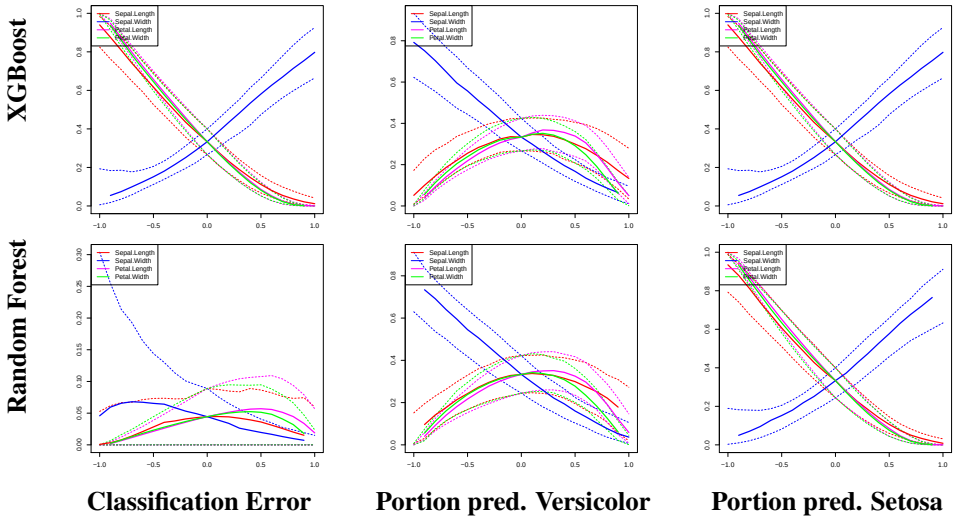

Figure 5: Evaluation of the classification error and the prediction with respect to the explanatory variable perturbation τ, on the *Iris* dataset. The quantity τ and the plain/dashed lines have the same signification as in the main part. **(Top)** XGBoost Model. The sepal width enables to differenciate the *Setosa* class. **(Bottom)** Random Forest Model. The sepal width again enables to differenciate the *Setosa* class.

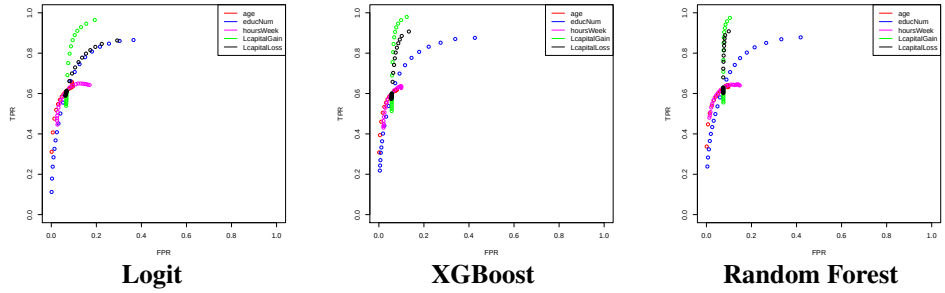

Figure 6: Evolution of Roc Curves in the *Adult income* dataset (Section 4.1). As for the classification errors, we observe that large values of the variable *hoursWeek* make the classification difficult.