Kernel Logistic Regression Approximation of an Understandable ReLU Neural Network

Marie Guyomard [1] **Susana Barbosa** [2] **Lionel Fillatre** [1]

Abstract

This paper proposes an understandable neural network whose score function is modeled as an additive sum of univariate spline functions. It extends usual understandable models like generative additive models, spline-based models, and neural additive models. It is shown that this neural network can be approximated by a logistic regression whose inputs are obtained with a non-linear preprocessing of input data. This preprocessing depends on the neural network initialization but this paper establishes that it can be replaced by a non random kernel-based preprocessing that no longer depends on the initialization. Hence, the convergence of the training process is guaranteed and the solution is unique for a given training dataset.

1. Introduction

Neural Networks (NNs) are widely used as their performance on both regression and classification tasks is significant (Meijering et al., 2022). The explainability of NNs and the possibility to certify their results is often criticized, which is why they are described as "black boxes" (Fel & Vigouroux, 2020).

In recent years, many studies have been conducted on the interpretability of the NNs, but also on the convergence of the learning process and the uniqueness of the learned parameters. A rigorous bridge between NNs using a rectified linear unit function (ReLU NNs) and spline-based multidimensional functions approximation has been built in (Balestriero et al., 2018). The authors establish that ReLU NNs can be interpreted as splines operators that divide the input space into polyhedrons. Unfortunately, the partition induced by the hidden layers of the network is too complex because the input variables are mixed with a linear transformation or convolution. Other methods like Decision Trees (DTs) (Breiman, 2017; Hastie et al., 2009), Multivariate Adaptative Regression Splines (MARS) models (Friedman, 1991) and Generalized Additive Models (GAMs) (Hastie, 2017) also partition the input space, while their partitions are interpretable. These partitions are only composed of orthotopes because they split each input variable separately. However, these methods exploit a greedy algorithm to split the variables at each iteration. Hence, both the optimality of the training and its robustness are questionable.

Recently, some studies were conducted to develop interpretable NN-based models that mimic these interpretable methods while optimizing a global criterion. In (Eckle et al., 2019), it is shown that the MARS model can be approximated by a NN but no algorithm is proposed to train it. The Neural Additive Models (NAMs) in (Agarwal et al., 2021) approximate GAMs by NNs. The resulting NN is basically a conventional NN learned with the Stochastic Gradient Descent (SGD) (Boyd & Vandenberghe, 2004) with no guarantee to converge toward a satisfying solution.

Recent papers as (Jacot et al., 2018) have demonstrated that it is possible to guarantee the convergence of the SGD for a fully connected NN when the number of hidden neurons becomes arbitrarily large. Indeed, assuming that the NN parameters are initialized with a Gaussian distribution, NNs may asymptotically become linear with respect to their parameters as NN width increases (Neal, 2012; Liu et al., 2020a). Unfortunately, it has been established in (Liu et al., 2020a) that this approximation does not hold when the output layer is non-linear, which arises for classification problems involving a sigmoid or a softmax function.

The contributions of this paper are fourfold. Firstly, we introduce an interpretable model that combines the advantages of MARS, GAMs, NAMs and NNs. Our Splines Approximation Throught Understandable ReLU Neural Network (SATURNN) is intrinsically explainable as MARS, GAM and NAM and its training involves to minimize a global classification loss. Secondly, we demonstrate that even if SATURNN is composed of a sigmoid output layer, it can be

[*]Equal contribution [1]Laboratory I3S, University Côte d'Azur, Sophia-Antipolis, France [2]Laboratory IPMC, University Côte d'Azur, Sophia-Antipolis, France. Correspondence to: Marie Guyomard <guyomard@i3s.unice.fr>, Lionel Fillatre <lionel.fillatre@i3s.unice.fr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

partially linearized with respect to its parameters when the number of hidden neurons is large. Thus its training can be done by solving a convex optimization problem. Thirdly, we show that SATURNN is equivalent to a logistic regression applied to non-linear features deduced from SATURNN architecture. Hence, with respect to the explainability taxonomy in (Linardatos et al., 2021; Xie et al., 2020), our structural explainability can be considered as global, model specific and intrinsically explainable. Finally, we show that this logistic regression is equivalent to a kernel logistic regression with a nonrandom kernel. Hence, we can conclude that the convergence of SATURNN is guaranteed and the obtained classifier is unique.

This paper is structured as follows. Section 2 introduces the SATURNN model. Section 3 demonstrates that the SATURNN model is equivalent to a logistic regression with non-linear features. Section 4 establishes an approximation of SATURNN by Kernel methods. Section 5 compares SATURNN and its approximation methods to state-of-the-art algorithms. Finally, Section 6 concludes the paper.

2. SATURNN Modeling

Our dataset contains N independent and identically distributed pairs $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ where $x^{(i)} \in \mathbb{R}^d$ is the vector of input variables or features and $y^{(i)} = \{0, 1\}$ is the binary label to predict. We suppose that the features lie on an open ball $\mathcal{B}_2^d(0, r)$ where $\mathcal{B}_2^d(c, r) := \{x \in \mathbb{R}^d : \|x - c\|_2 \leq r\}$ is the ball in $\mathbb{R}^d$ of center $c \in \mathbb{R}^d$ and radius $r > 0$ with $\|x\|_2^2 = \sum_{i=1}^{d} x_i^2$ the Euclidean norm of $x = (x_1, \ldots, x_d)$.

2.1. ReLU Neural Network

NNs (Goodfellow et al., 2016) are widely used for non-linear classification tasks, as they are universal approximators of any function (Hornik et al., 1989; Leshno et al., 1993). Let us consider a 1-hidden-ReLU-layer NN with p neurons and a sigmoid activation as output layer:

$$\Phi^{\text{ReLU(p)}}(x) = \sigma(\psi^{\text{ReLU(p)}}(x)), \tag{1}$$

$$\psi^{\text{ReLU(p)}}(x) = \beta_0 + \sum_{k=1}^{p} \beta_k \phi\left(W_k x + b_k\right), \tag{2}$$

with $\beta = (\beta_0, \beta_1, \ldots, \beta_p) \in \mathbb{R}^{p+1}$, $b = (b_1, \ldots, b_p) \in \mathbb{R}^p$ and $W_1, \ldots, W_p \in \mathbb{R}^{1\times d}$ the trainable parameters. The NN (1) models a non-linear classification as it applies the sigmoid function $\sigma(t) = 1/(1 + \exp(-t))$ to the non-linear score function (2). The function $\phi(\cdot) = \text{ReLU}(\cdot) = \max\{0, \cdot\}$ is the ReLU activation (Balestriero et al., 2018). The weight matrices W_k in (2) can be viewed as blenders as they mix all the input features. The resulting partition of a ReLU NN with 10 neurons is illustrated in Figure 1 (left). This partition with oblique regions is very difficult to interpret with respect to the input features.

2.2. SATURNN

This paper introduces the SATURNN model which is a classification ReLU NN (1) with the specific score function:

$$\psi(x, \theta) = \frac{1}{\sqrt{p}} \left[\beta_0 + \sum_{k=1}^{p} \beta_k \phi(s_k x_{\upsilon(k)} + b_k)\right], \tag{3}$$

where $\theta = [\beta^T, b^T]^T \in \mathbb{R}^{2p+1}$ are the trainable parameters and x^T is the transpose of x. Each of the p hidden neurons $h_k(x) = \phi(s_k x_{\upsilon(k)} + b_k)$ takes a single variable as an input, i.e., the k-th neuron $h_k(x)$ processes $x_{\upsilon(k)}$ where $\upsilon : \{1, \ldots, p\} \mapsto \{1, \ldots, d\}$ is the input selector indicating which feature is handled by the neuron. Since the ReLU function is non-decreasing, the sign $s_k \in \{-1, 1\}$ specifies if $h_k(x)$ is a non-decreasing or non-increasing function of $x_{\upsilon(k)}$. The s_k's are not estimated, they are randomly distributed as a Bernoulli distribution with the parameter $1/2$. Similarly, the selector function is randomly initialized such that $\upsilon(k)$ takes the value $i \in \{1, \ldots, d\}$ with the probability $1/d$, thus all the features are selected the same number of times on average. Finally, the bias b_k indicates the knot, which is the starting point when $h_k(x)$ becomes linear: $h_k(x) = s_k x_{\upsilon(k)} + b_k$ when $s_k x_{\upsilon(k)} > -b_k$. This construction of the hidden layer partitions the input space with orthotopes (Figure 1 right) leading to an interpretable decision rule. As an orthotope is the product of closed intervals, it is clear which variables are involved in this orthotope and what their range is. Of course, using orthotopes is not the only way to get an explainable architecture but we believe this is an efficient means to measure the role of each input variable in the classification decision (as it is done in GAMs, NAMs, MARS, decision trees, and so on).

It is worth noting that the SATURNN model can be seen as a special case of GAMs (Hastie, 2017). By rewriting (3), we get:

$$\psi(x, \theta) = \frac{1}{\sqrt{p}} \left[\beta_0 + \sum_{i=1}^{d} \underbrace{\sum_{1\leq k\leq p:\upsilon(k)=i} \beta_k \phi\left(s_k x_i + b_k\right)}_{f_i(x_i)}\right] \tag{4}$$

where $f_i(x_i)$ is the univariate spline function of x_i as a linear sum of ReLU functions. According to the taxonomy introduced by (Gilpin et al., 2018; Guidotti et al., 2018), the proposed method is intrinsically interpretable as it meets all the criteria including the explicability of both the outcome and the representation of the data inside the NN.

2.3. Loss function optimization and initialization

Contrary to NAMs (Agarwal et al., 2021) that optimize a criteria to learn a combination of $d + 1$ NNs, our proposed

method estimates the whole decision rule with only one NN. Learning SATURNN requires minimizing the following cost function:

$$\mathcal{L}^{\text{SATURNN}}(\theta) = \frac{1}{N}\sum_{i=1}^{N} L\left(\sigma(\psi(x^{(i)},\theta)), y^{(i)}\right), \quad (5)$$

such that

$$\hat{\theta}^{\text{SATURNN}} = \underset{\theta\in\mathcal{B}_2^{2p+1}(\theta^{(0)},R)}{\arg\min}\ \mathcal{L}^{\text{SATURNN}}(\theta), \quad (6)$$

with $L(\cdot)$ the loss function, for instance the binary cross-entropy used for binary classification tasks (Goodfellow et al., 2016; Murphy, 2013):

$$L(\hat{y}, y) = -y\log(\hat{y}) - (1-y)\log(1-\hat{y}), \quad (7)$$

for all $y \in \{0,1\}$ and all $0 \leq \hat{y} \leq 1$. As for all NN training, an initialization of the NN parameter is made with a vector $\theta^{(0)} = [\beta^{(0)^T}, b^{(0)^T}]^T$. The $\beta_k^{(0)}$'s satisfy $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ for $k = \{0,\dots,p\}$. The thresholds $b_k^{(0)}$ are initialized according to a uniform distribution on the interval $[-r, r]$ since $\|x_i\|_2 \leq r$: $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. A ℓ_2-regularization is applied to ensure that after the training, the estimated parameters $\hat{\theta}^{\text{SATURNN}}$ do not deviate too much from the initialized values, up to a distance maximum $R > 0$. Since $\psi(x,\theta)$ is non-linear on both x and θ, the optimization problem (6) is not convex. Hence, we have no guarantee about its convergence and the uniqueness of $\hat{\theta}^{\text{SATURNN}}$. In the next section, we will show that for a large p, the score function $\psi(x,\theta)$ becomes linear with respect to the NN parameter θ and thus the optimization problem (6) becomes convex.

3. Approximation with a Logistic Regression

As shown in (Liu et al., 2020a), a straightforward linearization of a sigmoid-based NN like (1) cannot be accurate. Hence, we first propose to linearize the score function $\psi(x,\theta)$ with respect to θ. Then, we do not linearize the sigmoid but we show that the resulting NN is equivalent to a Logistic Regression (LR) with non-linear features. It is important to note that we cannot use the recent results about parameterized NN (Liu et al., 2020a;b) because our SATURNN model does not satisfy the usual assumptions on the weight matrices W_k (our model does not contain such matrices) and the coefficients β_k. Our model is also specific because we explicitly take into account the bias vector b with a non-Gaussian random initialization. Hence, our theoretical results in Theorems 3.4 and 3.5 are novel.

3.1. Partial Linearization of SATURNN

We first proceed to an approximation of the score function (3) around its initial trainable parameters $\theta^{(0)}$. To do so, we consider fixed sample x and derive the second order Taylor series. We suppose that $\psi(x,\theta) : \mathbb{R}^d \to \mathbb{R}$ is $\mathcal{C}^2$ almost everywhere on the open ball $\mathcal{B}_2^{2p+1}(\theta^{(0)}, R) := \{\theta \in \mathbb{R}^{2p+1} : \|\theta - \theta^{(0)}\|_2 \leq R\}$ with fixed radius $R > 0$. Then there exists a real number $\tau \in [0,1]$ such that:

$$\tilde{\psi}(x,\theta,\theta^{(0)}) = \psi(x,\theta^{(0)}) + \nabla_\theta\psi(x,\theta^{(0)})^T(\theta-\theta^{(0)}) + \frac{1}{2}(\theta-\theta^{(0)})^T H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))(\theta-\theta^{(0)}), \quad (8)$$

where $\nabla_\theta\psi(x,\theta^{(0)})$ is the gradient vector of $\psi(x,\theta)$ with respect to θ taken at the point $\theta^{(0)}$ (Seber & Wild, 2005) defined by the following equation:

$$\nabla_\theta\psi(x,\theta^{(0)}) = \frac{1}{\sqrt{p}}\Big[1,\ \phi(s_1 x_{\upsilon(1)} + b_1^{(0)}), \phi(s_2 x_{\upsilon(2)} + b_2^{(0)}),\dots,\ \phi(s_p x_{\upsilon(p)} + b_p^{(0)}), \beta_1^{(0)}\mathbb{1}_{\{s_1 x_{\upsilon(1)} + b_1^{(0)} > 0\}},\ \dots,\ \beta_p^{(0)}\mathbb{1}_{\{s_p x_{\upsilon(p)} + b_p^{(0)} > 0\}}\Big]^T, \quad (9)$$

where $\mathbb{1}_A$ is the indicator function of the event A. In (8), $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))$ denotes the hessian matrix of $\psi(x,\theta)$ with respect to θ taken at the point $(1-\tau)\theta^{(0)} + \tau\theta$. The following lemma establishes that the gradient (9) remains constant as the width of the SATURNN p grows.

Lemma 3.1 (Constancy of the gradient vector)**.** *Let $\theta^{(0)} = [\beta_0^{(0)},\dots,\beta_p^{(0)}, b_1^{(0)},\dots b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. As the width of the SATURNN grows to infinity ($p \to \infty$), the gradient vector remains constant:*

$$\sup_{\substack{x\in\mathcal{B}_2^d(0,r)\\ \theta\in\mathcal{B}_2^{2p+1}(\theta^{(0)},R)}} \nabla_\theta\psi\left(x,\theta^{(0)}\right)^T\nabla_\theta\psi\left(x,\theta^{(0)}\right) = O(1), \quad (10)$$

where $O(\cdot)$ is the Big O notation.

Proof. The proof is provided in Appendix A. □

The following lemma establishes that the hessian matrix $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))$ approaches zero as the width of SATURNN p grows.

Lemma 3.2 (Asymptotic behavior of the hessian)**.** *Let $\theta^{(0)} = [\beta_0^{(0)},\dots,\beta_p^{(0)}, b_1^{(0)},\dots b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$, $b_k^{(0)} \sim \mathcal{U}[-r,+r]$ and $\tau \in [0,1]$. As the width of the SATURNN grows to infinity ($p \to \infty$), the hessian of $\psi(x,\theta)$ with respect to θ taken at the point $(1-\tau)\theta^{(0)} + \tau\theta^{(0)}$ approaches zero.*

$$\sup_{\substack{x\in\mathcal{B}_2^d(0,r)\\ \theta\in\mathcal{B}_2^{2p+1}(\theta^{(0)},R)}} \left\|H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\right\|_2 = O\left(\frac{1}{\sqrt{p}}\right). \quad (11)$$

Proof. The proof is provided in Appendix B. □

Thus, as the width of SATURNN grows, the second part of the Taylor development of the score function (8) is negligible (Lemma 3.2) and the gradient term remains constant (Lemma 3.1). We can then conclude in Proposition 3.3 that when SATURNN's width is large enough, its score function can be well-approximated by the linear model (12).

Proposition 3.3 (Linearization of $\psi(x,\theta)$)**.** *Let $\theta^{(0)} = [\beta_0^{(0)}, \dots, \beta_p^{(0)}, b_1^{(0)}, \dots b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. As the width of the NN grows to infinity ($p \to \infty$), the hessian matrix approaches zero while the gradient vector remains constant. Then the score function $\psi(x,\theta)$ can be well-approximated by the following linear model:*

$$\psi^{lin}(x,\theta,\theta^{(0)}) = \psi(x,\theta^{(0)}) + \nabla_\theta \psi(x,\theta^{(0)})^T(\theta - \theta^{(0)}). \tag{12}$$

Proof. The proof follows from Lemmas 3.1 and 3.2. □

It is a partial linearization because the function is still a non-linear function of x through the gradient

$$g_0(x) = \nabla_\theta \psi(x,\theta^{(0)}).$$

In fact, $g_0(x)$ does not depend on θ anymore but it is still a non-linear function of x. The following theorem proves that the approximation error between $\psi(x,\theta)$ defined in (3) and its linearized form $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ in (12) vanishes as p increases.

Theorem 3.4 (Approximation error of $\psi(x,\theta)$ by $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$)**.** *Let $\theta^{(0)} = [\beta_0^{(0)}, \dots, \beta_p^{(0)}, b_1^{(0)}, \dots, b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. Then, the approximation error between $\psi^{lin}(x,\theta,\theta^{(0)})$ and $\psi(x,\theta)$ is uniformly bounded by*

$$\sup_{\substack{x\in\mathcal{B}_2^d(0,r)\\ \theta\in\mathcal{B}_2^{2p+1}(\theta^{(0)},R)}} \left|\psi(x,\theta) - \psi^{lin}(x,\theta,\theta^{(0)})\right| \leq \frac{R^2}{2\sqrt{p}}. \tag{13}$$

Proof. The proof is provided in Appendix C. □

It is worth noting that the upper bound does not depend on x. The ℓ_2-regularization added in the cost function (6) through the constraint $\theta \in \mathcal{B}_2^{2p+1}(\theta^{(0)}, R)$ for a reasonable value of R ensures that the approximation error remains negligible when p is large enough. The following theorem shows that the sigmoid does not change the quality of the approximation when $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ replaces $\psi(x,\theta)$.

Theorem 3.5 (Local linearization of SATURNN)**.** *Let $\theta^{(0)} = [\beta_0^{(0)}, \dots, \beta_p^{(0)}, b_1^{(0)}, \dots, b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. Then, the approximation error between $\sigma(\psi(x,\theta))$ and $\sigma\left(\psi^{lin}(x,\theta,\theta^{(0)})\right)$ is uniformly bounded by*

$$\sup_{\substack{x\in\mathcal{B}_2^d(0,r)\\ \theta\in\mathcal{B}_2^{2p+1}(\theta^{(0)},R)}} \left|\sigma(\psi(x,\theta)) - \sigma(\psi^{lin}(x,\theta,\theta^{(0)}))\right| \leq \frac{R^2}{8\sqrt{p}}. \tag{14}$$

Proof. The proof is provided in Appendix D. □

According to Theorem 3.5, when the number p of neurons is large enough, the SATURNN model becomes equivalent to a LR model applied on the linearized form of the score function.

3.2. Equivalence with a Logistic Regression

In this subsection, we will demonstrate the equivalence between learning the SATURNN model by minimizing the cost function (6) and optimizing a LR based on $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$. Let $\delta_{LR}(x,\eta)$ denote the LR applied on the SATURNN linearized score function (12) defined by:

$$\delta_{LR}(x,\eta) = \sigma\left(c_0(x) + g_0(x)^T\eta\right) \approx \sigma\left(g_0(x)^T\eta\right) \tag{15}$$

where $\eta = \theta - \theta^{(0)} \in \mathbb{R}^{2p+1}$ is the parameter vector to learn when we train the LR. The non-linear feature mapping $g_0(x)$ can be considered as a non-linear preprocessing of the input feature x. The term $c_0(x) = \psi(x,\theta^{(0)})$ is constant with respect to η. It only depends on $\theta^{(0)}$, the initialization of the NN. A short calculation shows that the expectation of $c_0(x)$ is zero and its variance is bounded by $4r^2 + 1/p$. This variance is negligible with respect to $g_0(x)^T\eta$, which justifies the right-hand side approximation in (15). The vector of parameters η is estimated by minimizing

$$\mathcal{L}^{\text{LR}}(\eta) = \frac{1}{N}\sum_{i=1}^{N} L\left(\delta_{LR}(x^{(i)},\eta), y^{(i)}\right). \tag{16}$$

such that

$$\hat{\eta}^{\text{LR}} = \underset{\eta\in\mathcal{B}_2^{2p+1}(0,R)}{\arg\min}\ \mathcal{L}^{\text{LR}}(\eta). \tag{17}$$

Since $\delta_{LR}(x,\eta)$ in (15) is a usual linear logistic model learned with the cross-entropy loss (16) with ℓ_2-regularization over η, the optimization problem (17) is strongly convex. Hence, the convergence is guaranteed and the solution $\hat{\eta}^{\text{LR}}$ is unique. For any SATURNN initialization $\theta^{(0)}$, we get a unique solution $\hat{\eta}^{\text{LR}}\left(\theta^{(0)}\right)$.

As established in the following theorem, it is equivalent to optimize either (6) or (17) when the number p of neurons composing SATURNN is large enough.

Theorem 3.6 (Equivalence between $\mathcal{L}^{\text{SATURNN}}$ and $\mathcal{L}^{\text{LR}}$)**.** *Let $\theta^{(0)} = [\beta_0^{(0)}, \ldots, \beta_p^{(0)}, b_1^{(0)}, \ldots, b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. Then, we get*

$$\sup_{\substack{\theta \in \mathcal{B}_2^{2p+1}(\theta^{(0)},R) \\ \eta \in \mathcal{B}_2^{2p+1}(0,R)}} \left| \mathcal{L}^{SATURNN}(\theta) - \mathcal{L}^{LR}(\eta) \right| \leq \frac{R^2}{2\sqrt{p}}. \quad (18)$$

Proof. The proof is provided in Appendix E. □

According to the Theorem 3.6, learning $\hat{\theta}^{\text{SATURNN}}$ or $\hat{\eta}^{\text{LR}}$ leads to the same decision rule when p is large enough since i) their decision functions are almost equal according to Theorem 3.5 and ii) the training criteria are almost equal according to Theorem 3.6. Then, it is possible to deduce from $\delta_{\text{LR}}(x,\eta)$ both the partitioning of the input space induced by the SATURNN architecture but also the resulted estimated splines (4) by computing

$$\hat{\theta}^{\text{LR}} = \hat{\eta}^{\text{LR}} + \theta^{(0)}. \quad (19)$$

By approximating SATURNN with the LR $\delta_{\text{LR}}(x)$, we override the constraints making NNs qualified as "black boxes". On the one hand, the decision rule is interpretable as we obtain a partitioning of the input space with orthotopes and the estimated univariate functions can easily be computed providing information about the impact of each feature on the decision. On the other hand, the resulted decision rule is explainable since the LR offers guarantees on the convergence and the uniqueness of its estimate. It is worth noting that our LR model still depends on $\theta^{(0)}$ through $g_0(x)$. In the next section, we will introduce an approximation of $\delta_{\text{LR}}(x)$ using a kernel method that does not depend on the initializations $\theta^{(0)}$.

4. Approximation with a Kernel Logistic Regression

In (Jacot et al., 2018), the authors built a rigorous bridge between NNs and Kernel Methods. Nevertheless, this theory and those which followed later (Liu et al., 2020a;b) cannot be applied to SATURNN. In fact, (Jacot et al., 2018; Neal, 2012; Lee et al., 2017; Liu et al., 2020b) demonstrate that NNs whose parameters are initialized with Gaussian distributed random values are equivalent to Gaussian processes in infinite-width limit. Moreover, (Liu et al., 2020a) highlights that the tangent kernel does not remain constant when NNs are composed of a non-linear output layer. In this section, we will demonstrate that despite its particular architecture including the fixed weight matrix, the non-Gaussian initialization of the parameters, and its non-linear output layer, SATURNN can be well approximated by a Kernel Logistic Regression (KLR). First, we introduce the exact kernel that depends on the initialization $\theta^{(0)}$. Then, we establish that this kernel can be approximated by a constant kernel which is independent from the initial parameters, provided that the number of hidden neurons is large enough.

4.1. KLR approximation of $\delta_{\text{LR}}(x,\eta)$

According to the representer theorem (Schölkopf et al., 2001), the estimated vector of parameters $\hat{\eta}^{\text{LR}}$ in (17) can be expressed as a linear combination of the input vectors, i.e., there exist $\{\alpha_j\}_{j=1}^N \in \mathbb{R}^N$ such that

$$\hat{\eta}^{\text{LR}} = \sum_{j=1}^{N} \alpha_j g_0\left(x^{(j)}\right). \quad (20)$$

Let us denote $\kappa_0(x,\tilde{x})$ the kernel function defined by

$$\kappa_0(x,\tilde{x}) = g_0(x)^T g_0(\tilde{x}). \quad (21)$$

A short calculation shows that

$$\kappa_0(x,\tilde{x}) = \frac{1}{p}\left[1 + \sum_{k=1}^{p} \phi\left(s_k x_{v(k)} + b_k^{(0)}\right)\phi\left(s_k \tilde{x}_{v(k)} + b_k^{(0)}\right) + {\beta_k^{(0)}}^2 \mathbb{1}_{\{s_k x_{v(k)} + b_k^{(0)} > 0\}} \mathbb{1}_{\{s_k \tilde{x}_{v(k)} + b_k^{(0)} > 0\}}\right]. \quad (22)$$

It follows from (20) that (15) can be rewritten as

$$\delta_{KLR}(x,\alpha) = \sigma\left(\sum_{j=1}^{N} \alpha_j \kappa_0\left(x^{(j)}, x\right)\right). \quad (23)$$

Let us define the feature vector $K_0(x) \in \mathbb{R}^N$ as

$$K_0(x) = \left(\kappa_0\left(x^{(1)}, x\right), \ldots, \kappa_0\left(x^{(N)}, x\right)\right)^T. \quad (24)$$

Then, we get

$$\delta_{KLR}(x,\alpha) = \sigma\left(K_0(x)^T \alpha\right). \quad (25)$$

As for $\delta_{\text{LR}}(x,\eta)$ modeling, LR is applied on a non-linear feature mapping derived from the kernel. For $\delta_{\text{LR}}(x,\eta)$, the mapping of each sample $x \mapsto g_0(x) = \nabla_\theta \psi(x, \theta^{(0)})$ is independent from the training samples. When considering the kernel approach $\delta_{\text{KLR}}(x,\alpha)$ in (23), the mapping of each sample depends directly on the distribution of the whole training dataset : $x \mapsto \left(\kappa_0(x^{(i)}, x)\right)_{i=1}^N$. The univariate spline functions learned by SATURNN modeling, and thus the univariate segmentation, can be retrieved by computing

$$\tilde{\theta}^{\text{KLR}} = \sum_{j=1}^{N} \hat{\alpha}_j^{\text{KLR}} g_0(x^{(j)}) + \theta^{(0)}, \quad (26)$$

where $\hat{\alpha}^{\text{KLR}}$ minimizes

$$\mathcal{L}^{\text{KLR}}(\alpha) = \frac{1}{N}\sum_{i=1}^{N} L\left(\delta_{KLR}(x^{(i)},\alpha), y^{(i)}\right), \tag{27}$$

such that

$$\hat{\alpha}^{\text{KLR}} = \underset{\alpha\in\mathcal{B}_2^N(0,R)}{\arg\min}\ \mathcal{L}^{\text{KLR}}(\alpha). \tag{28}$$

Since KLR is based on the representer theorem applied to the LR, they are obviously equivalent.

4.2. Deterministic KLR approximation of $\delta_{\text{LR}}(x,\eta)$

By definition, the kernel $\kappa_0(x,\tilde{x})$ in (21) still depends on $\theta^{(0)}$, so does $K_0(x)$. However, the following proposition shows that this kernel can be well approximated by its expectation.

Proposition 4.1 (Convergence of the kernel)**.** *Let $\theta^{(0)} = [\beta_0^{(0)},\dots,\beta_p^{(0)},b_1^{(0)},\dots,b_p^{(0)}]$ and $r, R > 0$ such that $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r]$. The expectation of $\kappa_0(x,\tilde{x})$ with respect to $\theta^{(0)}$, for any couple $(x,\tilde{x}) \in \mathbb{R}^d \times \mathbb{R}^d$, is given by $\kappa(x,\tilde{x})$ defined as*

$$\kappa(x,\tilde{x}) = \frac{1}{p} + \frac{r^2}{6} + \frac{1}{4rd}\sum_{i=1}^{d} \varrho(x_i,\tilde{x}_i), \tag{29}$$

where $\varrho : \mathbb{R}^2 \mapsto \mathbb{R}$ is given by

$$\varrho(x_i,\tilde{x}_i) = 2r(x_i\tilde{x}_i + 1) - |x_i - \tilde{x}_i| + \frac{1}{6}|x_i - \tilde{x}_i|^3. \tag{30}$$

Furthermore, the variance $\mathbb{V}(\kappa_0(x,\tilde{x}))$ of $\kappa_0(x,\tilde{x})$ satisfies

$$\mathbb{V}(\kappa_0(x,\tilde{x})) = O\left(\frac{1}{p^2}\right). \tag{31}$$

Proof. The proof is given in Appendix F. □

From the law of large numbers, we can deduce that the kernel $\kappa_0(x,\tilde{x})$ converges in probability towards its expected value $\kappa(x,\tilde{x})$ as p goes to infinity. Hence, when the number of hidden neurons is large, we propose to approximate $\delta_{KLR}(x,\alpha)$ with $\delta_{EKLR}(x,\alpha)$ defined by

$$\delta_{EKLR}(x,\alpha) = \sigma\left(K(x)^T\alpha\right) \tag{32}$$

where the feature vector $K(x) \in \mathbb{R}^N$ is

$$K(x) = \left(\kappa\left(x^{(1)},x\right),\dots,\kappa\left(x^{(N)},x\right)\right)^T. \tag{33}$$

The Expected Kernel Logistic Regression (EKLR) does not depend anymore on $\theta^{(0)}$. Even though EKLR seems totally independent from SATURNN, it is possible to retrieve the decision rule estimated by SATURNN. Let $\hat{\alpha}^{\text{EKLR}}$ denote the parameters estimated by EKLR as the minimizer of (28) when δ_{EKLR} replaces δ_{KLR}. We get

$$\begin{aligned}\delta_{EKLR}(x,\hat{\alpha}^{\text{EKLR}}) &= \sigma\left(\sum_{j=1}^{N}\hat{\alpha}_j^{\text{EKLR}}\kappa\left(x^{(j)},x\right)\right)\\ &= \sigma\left(\hat{\beta}_0 + \frac{r^2}{4rd}\sum_{i=1}^{d}\hat{\beta}_i(x_i)\right)\end{aligned} \tag{34}$$

where

$$\hat{\beta}_0 = \frac{1}{p}\sum_{j=1}^{N}\hat{\alpha}_j^{\text{EKLR}}, \tag{35}$$

$$\hat{\beta}_i(x_i) = \sum_{j=1}^{N}\hat{\alpha}_j^{\text{EKLR}}\varrho(x_i^{(j)},x_i). \tag{36}$$

From (34), we retrieve an additive decomposition similar to SATURNN in (4). This decomposition does not depend on the initialization $\theta^{(0)}$. Each function $\hat{\beta}_i(x_i)$ depends on all the training samples $x^{(j)}$. Because of the ℓ_2-regularization to estimate the LR weights, the parameters $\hat{\alpha}_j^{\text{EKLR}}$ are unique. Hence, the functions $\hat{\beta}_i(x_i)$ are also unique for a given training dataset.

4.3. Implementation of KLR and EKLR

As mentioned previously, one of the main advantages of NNs and LR applied on a non-linear transformation of features concerns their optimization. Contrary to mainly other non-linear methods, as MARS (Friedman, 1991), GAM (Hastie, 2017) or DT (Breiman, 2017), these methods are learned using Stochastic Gradient Descent (Boyd & Vandenberghe, 2004). In (Zhu & Hastie, 2001) two algorithms are proposed to optimize Import Vector Machine, a classification approach built on KLR using classical kernels such as linear, polynomial or sigmoid, for example. Since the kernels proposed in (22) and (29) are novel, there is no implementation available in order to optimize both KLR (25) and EKLR (32) models. Thus, KLR and EKLR can be implemented as a regularized LR[1] applied on the preprocessed input sample $K_0(x)$ or $K(x)$.

5. Experiments

We evaluate the relevance of both the SATURNN model and its approximation methods $\delta_{\text{LR}}(x,\eta)$ in (15), $\delta_{\text{KLR}}(x,\alpha)$ in (23) and $\delta_{\text{EKLR}}(x,\alpha)$ in (32) on two simulated datasets and on one real one.

[1] Scikit-Learn Logistic Regression (Pedregosa et al., 2011).

5.1. Tested Methods

We compare the performance of our proposed methods to various non-linear models such as Random Forests (RF) (Breiman, 2001), MARS (Friedman, 1991), GAMs (Hastie, 2017), Explainable Boosting Machine (EBM) (Lou et al., 2012), ReLU NNs (Goodfellow et al., 2016) and NAMs (Agarwal et al., 2021). For all greedy algorithms, a tuning algorithm is required to learn the models with optimal hyperparameters. The optimal number of spline basis for MARS or the optimal width and number of DTs composing RFs and EBMs are computed with gridsearch algorithms. To get a fair comparison between all the methods, we do not incorporate interaction effects for MARS, GAM, EBM and NAMs. Tables 1 and 2 summarize the average accuracies of each model computed from a 5-fold cross-validation on both training and test samples. The computation time does not take into account the time required to tune the greedy methods, but rather the training of optimal parameterized models.

5.2. Simulated Datasets

We only consider two features in order to visualize both the estimated decision boundaries and the resulted partitions from different non-linear classification methods.

Gaussian Dataset. We generate 400 samples $x^{(i)} = (x_1^{(i)}, x_2^{(i)}) \in \mathbb{R}^2$ from a normal distribution. They are some instances of the pair of random values (X_1, X_2). In real applications the decision boundary is noisy, so we define the labels $y^{(i)}$ using a Bernoulli distribution:

$$y^{(i)} \sim \mathcal{B}\left(p(x^{(i)})\right) \text{ with } p(x^{(i)}) = \sigma\left(f_1(x_1^{(i)}) + f_2(x_2^{(i)})\right).$$

The probability used to generate the Bernoulli distribution is constructed from the sigmoid applied on a score function built with threshold effects. The functions f_1 and f_2 are defined to generate the regions displayed in Figure 1. We assume that having X_1 under a certain threshold decreases the Bernoulli probability, whereas a high value increases it. We also suppose that too low or too high of a level of X_2 increases the probability of belonging to class 2.

Circle Dataset. We generate two noisy circle distributions[2] each composed of 200 samples. The resulted dataset (Figure 4) is relevant to analyze non-linear classifiers.

Results. Although the SATURNN model does not mix the features but processes each input variable separately, it obtains similar performance to other methods, even higher for the Circle Dataset (90% accuracy on the test sample). The SATURNN modeling ensures an explainable partition of the input space (Figure 1 right) with orthotopes contrary to ReLU NN (Figure 1 left), which produce oblique regions that are impossible to interpret. Moreover, contrary to the ReLU NN, SATURNN can be viewed as a specific GAM (4) since the estimated splines by SATURNN are univariate and thus can be analyzed (Figure 2). The NAM (pink curves) and SATURNN (orange curves) models estimate more consistent splines with respect to the simulated pattern than MARS or GAM. Indeed, in Figure 1, we can see that when X_2 is high, most of the samples belong to class 1, justifying why NAM and SATURNN splines are decreasing from a certain value of X_2. Paradoxically, the estimated spline of X_2 by GAM (Figure 2 green curve) is always increasing. Thus the interpretability of GAM splines are not reliable, questioning the convergence of the GAM greedy algorithm.

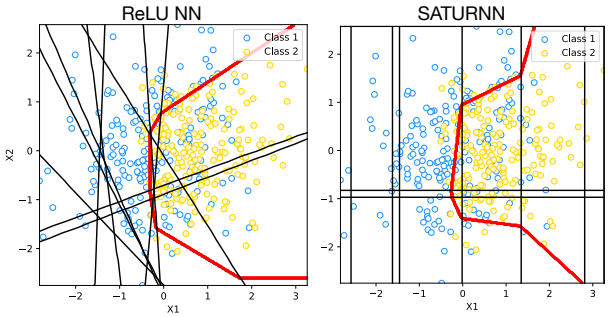

Figure 1. Partition on Gaussian Dataset of the ReLU NN ($p = 10$) on the left and SATURNN ($p = 10$) on the right. The estimated decision rule is in red and the partition boundaries are in black. *The estimated decision boundaries of all implemented methods are displayed in Figure 7 in Appendix G.1.*

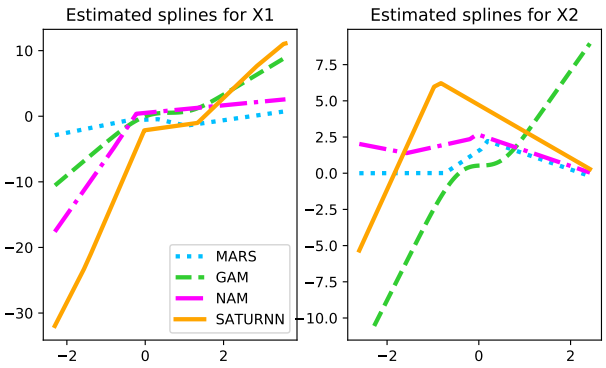

Figure 2. Gaussian Dataset: estimated splines for $X1$ (left) and $X2$ (right). *The estimated splines on the Circle Dataset are displayed in Figure 9 in Appendix G.2.*

As established in Theorem 3.5, the SATURNN model in its asymptotic regime (SATURNN$_\infty$ in Table 1 with respectively $p = 30\,000$ and $p = 50\,000$ for the Gaussian and the Circle Dataset) can be well-approximated by LR $\delta_{\text{LR}}(x, \eta)$ defined by (15) on both data sets. Indeed, in Table

[2] Using the make_circles method provided by Scikit-Learn.

Methods	Gaussian Dataset			Circle Dataset		
	Accuracy Train	Accuracy Test	Computation Time	Accuracy Train	Accuracy Test	Computation Time
RF	0.97 (0.01)	0.76 (0.01)	0.64	0.93 (0.01)	0.88 (0.02)	0.89
MARS	0.82 (0.01)	**0.80** (0.01)	0.29	0.89 (0.01)	0.88 (0.04)	0.09
GAM	0.81 (0.01)	0.78 (0.01)	0.11	0.90 (0.01)	0.89 (0.03)	0.04
EBM	0.82 (0.01)	0.78 (0.02)	0.30	0.93 (0.01)	0.89 (0.03)	0.17
NAM	0.77 (0.02)	0.76 (0.01)	227	0.87 (0.07)	0.86 (0.08)	210
ReLU NN	0.78 (0.02)	0.76 (0.02)	208	0.90 (0.01)	0.89 (0.01)	177
SATURNN	0.77 (0.02)	0.76 (0.03)	233	0.88 (0.01)	**0.90** (0.01)	218
SATURNN$_\infty$	0.78 (0.01)	0.77 (0.04)	387	0.91 (0.01)	**0.90** (0.01)	387
LR PSI LIN	0.86 (0.01)	**0.81** (0.04)	19	0.93 (0.01)	0.89 (0.02)	15
KLR	0.78 (0.02)	**0.80** (0.03)	0.05	0.89 (0.01)	0.89 (0.03)	0.05
EKLR	0.78 (0.02)	**0.80** (0.03)	0.05	0.89 (0.01)	0.89 (0.02)	0.05

Table 1. Summary of comparison on both datasets: average (standard deviation) accuracies obtained on training and test samples and average training time per fold (in seconds). The blue rows refer to the new approaches presented in this paper. ReLU NN and SATURNN are computed with $p = 10$. LR PSI LIN (meaning LR applied on the partially linearized score function of SATURNN$_\infty$), KLR and EKLR are all composed by respectively $p = 30\,000$ and $p = 50\,000$ neurons for the Gaussian and the Circle datasets.

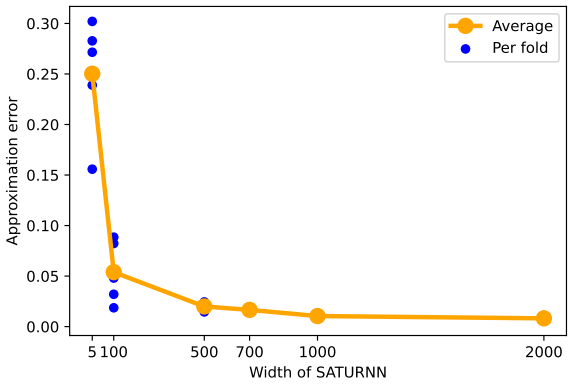

Figure 3. Gaussian Dataset: mean of the approximation errors E_k between SATURNN and $\delta_{\text{LR}}(x, \eta)$ over 5 folds for different values of p. *The results of this experiment on the circle dataset are displayed in Figure 10 in Appendix G.3.*

1, $\delta_{\text{LR}}(x, \eta)$ is as powerful as SATURNN and all the compared NNs and also much faster. Thus the decision rule of this approximation method fits the data well (Figure 4-left). Finally, we study the mean of the approximation error

$$E_k = \sum_{i \in \mathcal{F}_k} \left| \sigma(\psi(x^{(i)}, \hat{\theta})) - \sigma(\psi^{\text{lin}}(x^{(i)}, \hat{\theta}, \theta^{(0)})) \right| / |\mathcal{F}_k|$$

between SATURNN and $\delta_{\text{LR}}(x, \eta)$ over 5 folds (blue points) where $\mathcal{F}_k$ is the k-th fold, $|\mathcal{F}_k|$ its size and $\hat{\theta} = \hat{\theta}^{\text{SATURNN}}$ is obtained from (6). Figure 3 illustrates the mean of the approximation error (orange curve) over 5 folds (blue points) on the Gaussian Dataset (see Figure 10 in Appendix G.3 for the Circle Dataset). First, as p grows, the maximum value of the approximation error becomes more stable. Secondly, as established in Theorem 3.5, the mean of the approximation error decreases as $1/\sqrt{p}$.

Finally, the approximation of SATURNN by KLR and EKLR is much faster than NNs and obtain similar even higher accuracies on the test sample. For example, on the gaussian dataset, the accuracies on the test sample for both KLR and EKLR reach 80% while NAM and the ReLU NN get 76%. In Figure 4-right, the decision rule of the KLR method highlights how well the approximation works on the Circle Dataset.

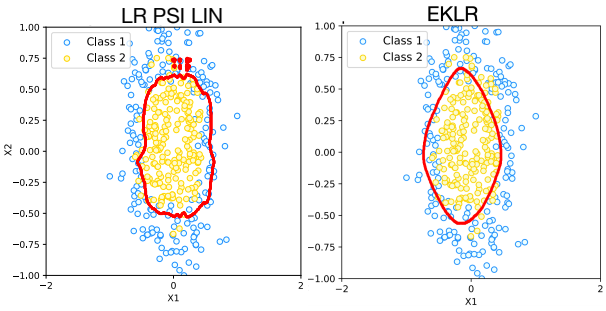

Figure 4. Estimated decision boundaries in red on the Circle Dataset of LR defined by (15) with $p = 50\,000$ on the left and EKLR (32) with $p = 50\,000$ on the right. *The estimated decision boundaries of all implemented methods are displayed in Figure 8 in Appendix G.1.*

5.3. Real dataset

We chose to derive experiments using the Framingham dataset (Mahmood et al., 2014). Indeed, the bio-medical field requires application of methods that can be interpreted by medical experts. The Framingham dataset aims to predict a cardiovascular event from 15 descriptive variables

	Train		Test		Computation
Methods	**Acc**	**AUC**	**Acc**	**AUC**	**Time**
RF	0.68 (0.01)	0.76 (0.01)	0.64 (0.02)	0.70 (0.02)	0.1
MARS	0.69 (0.01)	0.75 (0.01)	0.66 (0.02)	0.71 (0.01)	0.1
GAM	0.72 (0.01)	0.80 (0.01)	0.63 (0.02)	0.69 (0.02)	0.65
EBM	0.69 (0.01)	0.77 (0.01)	0.65 (0.01)	0.72 (0.01)	7.8
NAM	0.71 (0.01)	0.78 (0.01)	0.65 (0.01)	0.70 (0.02)	694
RN ReLU	0.75 (0.02)	0.85 (0.02)	0.61 (0.03)	0.66 (0.03)	483
SATURNN	0.68 (0.02)	0.74 (0.02)	**0.67 (0.03)**	0.72 (0.02)	735
SATURNN$_\infty$	0.63 (0.02)	0.72 (0.01)	0.62 (0.02)	0.71 (0.01)	591
RL PSI LIN	0.76 (0.01)	0.84 (0.01)	0.64 (0.02)	0.69 (0.02)	52
KLR	0.69 (0.01)	0.74 (0.01)	0.66 (0.02)	**0.73 (0.02)**	0.32
EKLR	0.68 (0.01)	0.74 (0.01)	0.66 (0.01)	**0.73 (0.02)**	0.35

Table 2. Summary of comparison on Framingham dataset: average accuracy and AUC (standard deviation) on the training and test samples and average training time (in seconds). The blue rows refer to the new approaches presented in this paper. The ReLU NN is composed of $p = 10$ neurons while SATURNN and NAM are trained with $p = 40$. KLR, EKLR and RL PSI LIN (LR applied to the linearized score function of the SATURNN$_\infty$) have $p = 50\,000$ neurons.

such as gender, cholesterol level, or body mass index. With the proportions of the database being strongly unbalanced (85% - 15%), we chose to rebalance it. We thus carry out the experiments on 1114 patients.

AUC (Area Under the Curve) is a criterion appreciated by doctors (Zhou et al., 2009). Where accuracy provides information about the predictive power of a model, AUC gives an idea of the Sensibility and the Specificity of a predictive model, that is to say the capability of a tool for identifying a binary signal (sick or not in our case). SATURNN produces similar, or even higher accuracies than the other methods: the SATURNN composed of $p = 40$ neurons reaches 72% of AUC on the validation sample, GAM 69% and NAM 70%. The greedy methods such as RF, MARS, and GAM tend to overfit. By mixing all the variables together, the ReLU NN does not allow us to graphically visualize the partitioning of the input space. In Figure 5, we can analyze the segmentations induced by the interpretable methods. NAM (pink curve) and SATURNN (orange) estimate similar splines for cigarettes per day and glucose. GAM (green) and MARS (blue) introduce few non-linearities, whereas their performance could have been improved as demonstrated by the AUC of SATURNN (Table 2). These difficulties are related to the fact that these methods do not optimize a global criterion but use an iterative method to add threshold effects. We can see that the SATURNN-induced spline for the diabetes variable (Figure 5-right) is zero. This behavior can be likened to variable selection: SATURNN estimates that the variables diabetes, heart rate per minute, and presence of cough have no significant effects on the development of the pathology. Although by excluding these variables from the decision rule, SATURNN performs better than NAM and GAM methods, which add complexity by thresholding these variables. Finally, the SATURNN approximation methods are much faster than those of the NNs (resp. 32 and 35 seconds for KLR and EKLR) and achieve better predictive performance (73% AUC on the validation sample for KLR and EKLR).

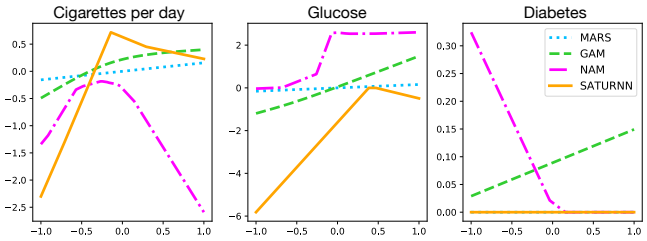

Figure 5. Estimated Splines on the Framingham dataset for MARS (blue), GAM (green), NAM (pink) and SATURNN with $p = 40$ neurons (orange).

6. Conclusion

This paper proposes an understandable ReLU neural network architecture based on a sum of univariate spline functions. It is shown that this network is asymptotically equivalent to a kernel logistic regression with a novel kernel as the number of hidden neurons becomes large. We finally get an additive model composed of a sum of univariate functions that depend only on the training samples. The convergence is guaranteed and the spline decomposition is unique.

References

Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., and Hinton, G. E. Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34: 4699–4711, 2021.

Balestriero, R. et al. A spline theory of deep learning. In *International Conference on Machine Learning*, pp. 374–383. PMLR, 2018.

Boyd, S. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, March 2004.

Breiman, L. Random forests. *Machine learning*, 45(1): 5–32, 2001.

Breiman, L. *Classification and regression trees*. Routledge, 2017.

Eckle, K. et al. A comparison of deep networks with relu activation function and linear spline-type methods. *Neural Networks*, 110:232–242, 2019.

Fel, T. and Vigouroux, D. Representativity and consistency measures for deep neural network explanations. *arXiv preprint arXiv:2009.04521*, 2020.

Friedman, J. H. Multivariate adaptive regression splines. *The annals of statistics*, 19(1):1–67, 1991.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE, 2018.

Goodfellow, I. J., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Hastie, T. J. Generalized additive models. In *Statistical models in S*, pp. 249–307. Routledge, 2017.

Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

Linardatos, P., Papastefanopoulos, V., and Kotsiantis, S. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2021.

Liu, C., Zhu, L., and Belkin, M. On the linearity of large non-linear models: when and why the tangent kernel is constant. *Advances in Neural Information Processing Systems*, 33:15954–15964, 2020a.

Liu, C., Zhu, L., and Belkin, M. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. *arXiv preprint arXiv:2003.00307*, 2020b.

Lou, Y., Caruana, R., and Gehrke, J. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 150–158, 2012.

Mahmood, S. S., Levy, D., Vasan, R. S., and Wang, T. J. The framingham heart study and the epidemiology of cardiovascular disease: a historical perspective. *The lancet*, 383(9921):999–1008, 2014.

Meijering, E., Calhoun, V. D., Menegaz, G., Miller, D. J., and Ye, J. C. Deep learning in biological image and signal processing. *IEEE Signal Processing Magazine*, 39 (2):24–26, 2022. doi: 10.1109/MSP.2021.3134525.

Murphy, K. P. *Machine learning : a probabilistic perspective*. MIT Press, Cambridge, 2013.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Schölkopf, B., Herbrich, R., and Smola, A. J. A generalized representer theorem. In Helmbold, D. and Williamson, B. (eds.), *Computational Learning Theory*, pp. 416–426. Springer Berlin Heidelberg, 2001.

Seber, G. and Wild, C. *Nonlinear Regression*. Wiley Series in Probability and Statistics. Wiley, 2005. ISBN 9780471725305. URL https://books.google.fr/books?id=YBYlCpBNo_cC.

Xie, N., Ras, G., van Gerven, M., and Doran, D. Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*, 2020.

Zhou, X.-H., McClish, D. K., and Obuchowski, N. A. *Statistical methods in diagnostic medicine*. John Wiley & Sons, 2009.

Zhu, J. and Hastie, T. Kernel logistic regression and the import vector machine. *Advances in neural information processing systems*, 14, 2001.

A. Proof of Lemma 3.1

Proof. The gradient vector of the SATURNN score function $\psi(x,\theta)$ with respect to θ taken at the point $\theta^{(0)}$ becomes constant as the width of the SATURNN growths.

Let us recall that the features x lie on an open ball $\mathcal{B}_2^d(0,r),\ r>0$ and the vector of initialized parameters is defined as $\theta^{(0)}=[\beta_0^{(0)},\beta_1^{(0)},\dots,\beta_p^{(0)},b_1^{(0)},\dots,b_p^{(0)}]$ with $\beta_k^{(0)}\sim\mathcal{N}(0,1)$ and $b_k^{(0)}\sim\mathcal{U}[-r,+r]$. We suppose $s_k\in\{-1,1\}\sim\mathcal{B}(1/2)$ and $\upsilon(k)\sim\mathcal{U}[\![1,d]\!]$, for each $k\in\{1,\dots,p\}$. We assume that $\psi(x,\theta)$ in (3) is $\mathcal{C}^2$ almost everywhere on an open ball $\mathcal{B}_2^{2p+1}:=\{\theta\in\mathbb{R}^{2p+1}:\|\theta-\theta^{(0)}\|_2\leq R\}$ with fixed radius $R>0$. In the following, we neglect the points where the function are not $\mathcal{C}^2$.

The gradient vector $\nabla_\theta\psi(x,\theta)$ is the gradient vector of $\psi(x,\theta)$ with respect to θ and is defined by:

$$\nabla_\theta\psi(x,\theta)=\left[\frac{\partial\psi(x,\theta)}{\partial\beta_0},\frac{\partial\psi(x,\theta)}{\partial\beta_1},\dots,\frac{\partial\psi(x,\theta)}{\partial\beta_p},\frac{\partial\psi(x,\theta)}{\partial b_1},\dots,\frac{\partial\psi(x,\theta)}{\partial b_p}\right]^T. \tag{37}$$

We have for all $k\in\{1,\dots,p\}$:

$$\begin{aligned}
\frac{\partial\psi(x,\theta)}{\partial\beta_0}&=\frac{1}{\sqrt{p}}\\
\frac{\partial\psi(x,\theta)}{\partial\beta_k}&=\frac{1}{\sqrt{p}}\phi(s_kx_{\upsilon(k)}+b_k)\\
\frac{\partial\psi(x,\theta)}{\partial b_k}&=\frac{1}{\sqrt{p}}\beta_k\frac{\partial\phi(s_kx_{\upsilon(k)}+b_k)}{\partial b_k}.
\end{aligned}$$

So we can rewrite (37) as:

$$\nabla_\theta\psi(x,\theta)=\left[\frac{1}{\sqrt{p}},\frac{1}{\sqrt{p}}\phi(s_1x_{\upsilon(1)}+b_1),\dots,\frac{1}{\sqrt{p}}\phi(s_px_{\upsilon(p)}+b_p),\frac{1}{\sqrt{p}}\beta_1\frac{\partial\phi(s_1x_{\upsilon(1)}+b_1)}{\partial b_1},\dots,\frac{1}{\sqrt{p}}\beta_p\frac{\partial\phi(s_px_{\upsilon(p)}+b_p)}{\partial b_p}\right]^T. \tag{38}$$

According to our assumptions, we have $\forall k\in\{1,\dots,p\}$, $b_k^{(0)}\in[-r,+r]$ and $s_kx_{\upsilon(k)}\in[-r,+r]$. Thus, $s_kx_{\upsilon(k)}+b_k^{(0)}\in[-2r,2r]$ and $\phi(s_kx_{\upsilon(k)}+b_k^{(0)})\in[0,2r]$.

We compute the norm of the gradient taken at the point $\theta^{(0)}$ in order to study its constancy:

$$\begin{aligned}
\langle\nabla_\theta\psi(x,\theta^{(0)}),\nabla_\theta\psi(x,\theta^{(0)})\rangle&=\nabla_\theta\psi(x,\theta^{(0)})^T\nabla_\theta\psi(x,\theta^{(0)})\\
&=\frac{1}{p}\Bigg[1+\underbrace{\sum_{i=1}^{p}\phi(s_ix_{\upsilon(i)}+b_i^{(0)})^T\phi(s_ix_{\upsilon(i)}+b_i^{(0)})}_{\in[0,4pr^2]}\\
&\quad+\underbrace{\sum_{i=1}^{p}{\beta_i^{(0)}}^2\mathbb{1}_{\{s_ix_{\upsilon(i)}+b_i^{(0)}>0\}}\mathbb{1}_{\{s_ix_{\upsilon(i)}+b_i^{(0)}>0\}}}_{\in[0,p\max(\beta)^2]}\Bigg].\\
&\leq\frac{1}{p}[1+4pr^2+p\max(\beta)^2]\\
&=O\left(\frac{1}{p}[O(1+4pr^2+p\max(\beta)^2)]\right)\\
&=O\left(\frac{1}{p}O(p)\right)\\
&=O(1).
\end{aligned} \tag{39}$$

□

B. Proof of Lemma 3.2

Proof. The hessian matrix of $\psi(x,\theta)$ with respect to θ taken at the point $(1-\tau)\theta^{(0)}+\tau\theta$ approaches zero as p growths.

Let us recall that the features x lie on an open ball $\mathcal{B}_2^d(0,r),\ r>0$ and the vector of initialized parameters is defined as $\theta^{(0)}=[\beta_0^{(0)},\beta_1^{(0)},\ldots,\beta_p^{(0)},b_1^{(0)},\ldots,b_p^{(0)}]$ with $\beta_k^{(0)}\sim\mathcal{N}(0,1)$ and $b_k^{(0)}\sim\mathcal{U}[-r,+r]$. We suppose $s_k\in\{-1,1\}\sim\mathcal{B}(1/2)$ and $\upsilon(k)\sim\mathcal{U}[\![1,d]\!]$, for each $k\in\{1,\ldots,p\}$. We assume that $\psi(x,\theta)$ in (3) is $\mathcal{C}^2$ almost everywhere on an open ball $\mathcal{B}_2^{2p+1}:=\{\theta\in\mathbb{R}^{2p+1}:\|\theta-\theta^{(0)}\|_2\leq R\}$ with fixed radius $R>0$.

The hessian matrix of the SATURNN score function $\psi(x,\theta)$ can be written as the following structure:

$$H_\theta(\psi(x,\theta))=\begin{pmatrix} H^{(1,1)} & H^{(1,2)} & \ldots & H^{(1,2p+1)} \\ H^{(2,1)} & H^{(2,2)} & \ldots & H^{(2,2p+1)} \\ \vdots & \vdots & \ddots & \vdots \\ H^{(2p+1,1)} & H^{(2p+1,2)} & \ldots & H^{(2p+1,2p+1)} \end{pmatrix}. \tag{40}$$

Here each hessian block $H^{(i,j)}:=\frac{\partial^2\psi(x,\theta)}{\partial\theta_i\partial\theta_j}$, $i,j\in\{1,\ldots,2p+1\}$, is the second derivative of $\psi(x,\theta)$ with respect to its parameters $\theta=[\beta_0,\beta_1,\ldots,\beta_p,b_1,\ldots,b_p]$, such that $\theta_0=\beta_0$ and $\theta_{2p+1}=b_p$ for example.

We derive the different second derivatives of $\psi(x,\theta)$ with respect to θ at the point $(1-\tau)\theta^{(0)}+\tau\theta$. We have for $i,j\in\{1,\ldots,p\}^2,\ i\neq j$ and $\phi(\cdot)=\text{ReLU}(\cdot)=\max\{0,\cdot\}$:

$$\begin{aligned}
&\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial^2\beta_0}=\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial^2\beta_j}=\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial\beta_i\partial\beta_j}=0,\\
&\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial b_i\partial b_j}=\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial\beta_i\partial b_j}=0,\\
&\frac{\partial^2\psi\left(x,(1-\tau)\theta^{(0)}+\tau\theta\right)}{\partial^2 b_j}=\frac{1}{\sqrt{p}}\beta_j\frac{\partial^2\phi\left(s_jx_{\upsilon(j)}+(1-\tau)b_j^{(0)}+\tau b_j\right)}{\partial^2 b_j}=0,\\
&\frac{\partial^2\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)}{\partial\beta_j\partial b_j}=\frac{\tau}{\sqrt{p}}\frac{\partial\phi\left(s_jx_{\upsilon(j)}+(1-\tau)b_j^{(0)}+\tau b_j\right)}{\partial b_j}=\frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_jx_{\upsilon(j)}+(1-\tau)b_j^{(0)}+\tau b_j>0\right\}}.
\end{aligned} \tag{41}$$

Finally, the hessian matrix $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))$ can be rewritten as:

$$\left(\begin{array}{c|ccc|ccc}
0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 \\ \hline
\vdots & & & & \ddots & & 0 \\
\vdots & & 0 & & & \frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_kx_{\upsilon(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}} & \\
\vdots & & & & 0 & & \ddots \\ \hline
\vdots & \ddots & & 0 & & & \\
\vdots & & \frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_kx_{\upsilon(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}} & & & 0 & \\
0 & 0 & & \ddots & & &
\end{array}\right). \tag{42}$$

We study the asymptotic comportment of (42) by computing its spectral norm :

$$\begin{aligned}\|H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\|_2 &= \sqrt{\lambda_{\max}\left[H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))^T H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\right]}\\ &= \sqrt{\lambda_{\max}\left[H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\right]^2},\ (H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta)) \text{ symmetric})\\ &= \lambda_{\max}\left[H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\right]. \end{aligned} \tag{43}$$

We derive the eigenvalues of the hessian matrix by computing $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))u = \lambda u$, with $u = [0,\dots,0,1,0,\dots,0,\alpha,0,\dots,0]^T \in \mathbb{R}^{2p+1}$. We obtain the following linear system of equations with two unknown parameters (λ,α):

$$\begin{cases} \lambda = 0, \\ \lambda = \alpha\frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}, \\ \alpha\lambda = \frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}. \end{cases}$$

By omitting the $2p+1$ solutions $\lambda = 0$, we resolve the above system of two equations and find:

$$\begin{aligned} &\lambda = \left(\frac{\tau^2}{\sqrt{p}}\right)^2 \frac{1}{\lambda}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}\\ &\Leftrightarrow \lambda^2 = \frac{1}{p}\tau^4\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}\\ &\Leftrightarrow \lambda = \pm\frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}\\ &\Leftrightarrow \lambda = \frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b^{(0)}+\tau b_k>0\right\}},\ \text{since } p\in\mathbb{R}^+. \end{aligned} \tag{44}$$

By shifting the non-zeros entries of u to positions k and $k+p$, for $k\in\{1,\dots,p\}$, we recover $2p$ eigenvalues. As the Hessian (42) is a matrix with $2p$ non-zero entries, we thus recover all eigenvalues and conclude that pairs of eigenvalues are given by:

$$\lambda_k, \lambda_{k+p} = \frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}. \tag{45}$$

The spectral norm of a matrix is the largest eigenvalue of the matrix :

$$\begin{aligned}\|H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))\|_2 &= \max_{k=\{1,\dots,p\}}\frac{\tau^2}{\sqrt{p}}\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}\\ &= \frac{1}{\sqrt{p}}\max_{k=\{1,\dots,p\}}\tau^2\mathbb{1}_{\left\{s_k x_{v(k)}+(1-\tau)b_k^{(0)}+\tau b_k>0\right\}}\\ &= \frac{1}{\sqrt{p}}\\ &= O\left(\frac{1}{\sqrt{p}}\right). \end{aligned} \tag{46}$$

□

C. Proof of Theorem 3.4

Proof. The error of approximation of the SATURNN score function $\psi(x,\theta)$ by its linearized form $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ is bounded by $\frac{R^2}{2\sqrt{p}}$ and thus tends to zero as SATURNN width increases.

Let us recall that the features x lie on an open ball $\mathcal{B}_2^d(0,r),\ r>0$ and the vector of initialized parameters is defined as $\theta^{(0)}=[\beta_0^{(0)},\beta_1^{(0)},\ldots,\beta_p^{(0)},b_1^{(0)},\ldots,b_p^{(0)}]$ with $\beta_k^{(0)}\sim\mathcal{N}(0,1)$ and $b_k^{(0)}\sim\mathcal{U}[-r,+r]$. We assume that $\psi(x,\theta)$ in (3) is $\mathcal{C}^2$ almost everywhere on an open ball $\mathcal{B}_2^{2p+1}:=\{\theta\in\mathbb{R}^{2p+1}:\|\theta-\theta^{(0)}\|_2\leq R\}$ with fixed radius $R>0$.

According to the Taylor series expansion theory, there exists $\tau\in[0,1]$ such that $\psi(x,\theta)$ can be exactly approximated by:

$$\psi(x,\theta,\theta^{(0)})=\psi(x,\theta^{(0)})+\nabla_\theta\psi(x,\theta^{(0)})^T(\theta-\theta^{(0)})+\frac{1}{2}(\theta-\theta^{(0)})^T H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))(\theta-\theta^{(0)}), \tag{47}$$

where $\nabla_\theta\psi(x,\theta^{(0)})$ is the gradient vector of $\psi(x,\theta^{(0)})$ with respect to θ taken at the point $\theta^{(0)}$. $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))$ denotes the hessian matrix of $\psi(x,\theta)$ with respect to its parameters θ taken at the point $(1-\tau)\theta^{(0)}+\tau\theta$.

According to proposition 3.3, we can rewrite the second order Taylor expansion series as follows:

$$\psi(x,\theta,\theta^{(0)})=\psi^{\text{lin}}(x,\theta,\theta^{(0)})+\frac{1}{2}(\theta-\theta^{(0)})^T H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))(\theta-\theta). \tag{48}$$

Thus the error of approximation of $\psi(x,\theta)$ by $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ is defined by:

$$\left|\psi(x,\theta)-\psi^{\text{lin}}(x,\theta,\theta^{(0)})\right|=\left|-\frac{1}{2}(\theta-\theta^{(0)})^T H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))(\theta-\theta^{(0)})\right|. \tag{49}$$

Since the hessian matrix $H_\theta(\psi(x,(1-\tau)\theta^{(0)}+\tau\theta))$ admits eigenvalues, see (45), and eigenvectors, it is diagonalizable by $P^T\Lambda P$ with $\Lambda=\text{diag}(\lambda_1,\ldots,\lambda_{2p+1})$ the diagonal matrix of eigenvalues and P the orthogonal matrix composed of associated eigenvectors. We can rewrite the error of approximation (49) as follows:

$$\begin{aligned}
\left|\psi(x,\theta)-\psi^{\text{lin}}(x,\theta,\theta^{(0)})\right| &= \left|-\frac{1}{2}(\theta-\theta^{(0)})^T P^T\Lambda P(\theta-\theta^{(0)})\right| \\
&= \left|-\frac{1}{2}z^T\Lambda z\right|,\ \text{with } z=P(\theta-\theta^{(0)}) \\
&= \left|-\frac{1}{2}\sum_{i=1}^{2p+1}\lambda_i z_i^2\right| \\
&\leq \left|-\frac{\lambda_{\max}}{2}\sum_{i=1}^{2p+1}z_i^2\right| \\
&\leq \left|-\frac{\lambda_{\max}}{2}\|z_i\|^2\right|.
\end{aligned}$$

We know that $\|\theta-\theta^{(0)}\|^2=\|P(\theta-\theta^{(0)})\|^2=\|z\|^2$. Since $\|\theta-\theta^{(0)}\|\leq R$, we have $\|z\|^2\leq R^2$. Thus we get:

$$\begin{aligned}
\left|\psi(x,\theta)-\psi^{\text{lin}}(x,\theta,\theta^{(0)})\right| &\leq \left|-\frac{\lambda_{\max}}{2}R^2\right| \\
&\leq \frac{R^2}{2\sqrt{p}} \quad \text{as} \quad \lambda_{\max}=\frac{1}{\sqrt{p}}>0,\ R^2>0 \qquad (50)\\
&= O\left(\frac{1}{\sqrt{p}}\right). \qquad (51)
\end{aligned}$$

□

D. Proof of Theorem 3.5

Proof. The approximation error of the SATURNN model $\sigma(\psi(x,\theta))$ by its partially linearized form $\sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}))$ is bounded by $\frac{R^2}{8\sqrt{p}}$ and thus tends to zero as SATURNN width increases.

Let us recall that the features x lie on an open ball $\mathcal{B}_2^d(0,r),\ r>0$. We assume that $\psi(x,\theta)$ in (3) is $\mathcal{C}^2$ almost everywhere on the open ball $\mathcal{B}_2^{2p+1} := \{\theta \in \mathbb{R}^{2p+1} : \|\theta - \theta^{(0)}\|_2 \leq R\}$ with fixed radius $R>0$. We know from Proposition 3.3 that we can approximate $\psi(x,\theta)$ by:

$$\psi(x,\theta) = \psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \epsilon(x,\theta,\theta^{(0)}), \tag{52}$$

with $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ defined in Proposition 3.3 by (12) and $\epsilon(x,\theta,\theta^{(0)})$ the approximation error bounded by $\frac{R^2}{2\sqrt{p}}$ as established in Theorem 3.4.

The first order Taylor expansion theory guarantees that there exists a $\tau \in [0,1]$ such that $\sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}))$ can be rewritten as:

$$\sigma\left(\psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \epsilon(x,\theta,\theta^{(0)})\right) = \sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)})) + \sigma'(\psi^{\text{lin}}(x,\theta,\theta^{(0)} + \tau\epsilon(x,\theta,\theta^{(0)})))\epsilon(x,\theta,\theta^{(0)})), \tag{53}$$

with $\sigma'(\cdot) = \sigma(\cdot)[1-\sigma(\cdot)]$ the first derivative of the sigmoid that is bounded by $\frac{1}{4}$.

The approximation error of $\sigma(\psi(x,\theta)) = \sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \epsilon(x,\theta,\theta^{(0)}))$ by $\sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ is equal to:

$$\begin{aligned}
\left|\sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \epsilon(x,\theta,\theta^{(0)})) - \sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}))\right| &= \left|\sigma'(\psi^{\text{lin}}(x,\theta) + \tau\epsilon(x,\theta,\theta^{(0)}))\epsilon(x,\theta,\theta^{(0)})\right| \\
&= \Big|\sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \tau\epsilon(x,\theta,\theta^{(0)})) \\
&\quad \times \left[1 - \sigma(\psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \tau\epsilon(x,\theta,\theta^{(0)}))\right]\epsilon(x,\theta,\theta^{(0)})\Big| \\
&\leq \left|\frac{1}{4}\epsilon(x,\theta,\theta^{(0)})\right| \\
&\leq \frac{R^2}{8\sqrt{p}} \qquad (54) \\
&= O\left(\frac{1}{\sqrt{p}}\right). \qquad (55)
\end{aligned}$$

□

E. Proof of Theorem 3.6

Proof. As the width of SATURNN growths, it becomes equivalent to learn the SATURNN model or the Logistic Regression applied on its linearized score function.

Let us recall that the N features $x^{(i)}$ lie on an open ball $\mathcal{B}_2^d(0,r),\ r>0$, $\sigma(\psi(x,\theta))$ denote the SATURNN model, with $\sigma(\cdot)$ the sigmoïd. We assume that the SATURNN score function $\psi(x,\theta)$, defined in (3), is $\mathcal{C}^2$ almost everywhere on an open ball $\mathcal{B}_2^{2p+1} := \{\theta \in \mathbb{R}^{2p+1} : \|\theta-\theta^{(0)}\|_2 \leq R\}$ with fixed radius $R>0$. $\psi^{\text{lin}}(x,\theta,\theta^{(0)})$ refers to the linearized score function defined by:

$$\psi^{\text{lin}}(x,\theta,\theta^{(0)}) = \psi(x,\theta^{(0)}) + g_0(x)^T(\theta-\theta^{(0)}),$$

with $g_0(x) = \nabla_\theta \psi(x,\theta^{(0)})$ the gradient of $\psi(x,\theta)$ with respect to θ taken at the point $\theta^{(0)}$ defined by (9). The vector of initialized parameters is defined as $\theta^{(0)} = [\beta_0^{(0)},\beta_1^{(0)},\dots,\beta_p^{(0)},b_1^{(0)},\dots,b_p^{(0)}]$ with $\beta_k^{(0)} \sim \mathcal{N}(0,1)$ and $b_k^{(0)} \sim \mathcal{U}[-r,+r],\ r>0$.

Let us study

$$\begin{aligned}\left|\mathcal{L}^{\text{SATURNN}}(\theta) - \mathcal{L}^{\text{LR}}(\eta)\right| &= \left|\frac{1}{N}\sum_{i=1}^{N} L\left(\sigma\left(\psi(x^{(i)},\theta)\right),y^{(i)}\right) - \sum_{i=1}^{N}\frac{1}{N}L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)},\theta,\theta^{(0)})\right),y^{(i)}\right)\right| \\ &= \left|\frac{1}{N}\sum_{i=1}^{N} L\left(\sigma\left(\psi(x^{(i)},\theta)\right),y^{(i)}\right) - L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)},\theta,\theta^{(0)})\right),y^{(i)}\right)\right|. \end{aligned} \tag{56}$$

From Theorem 3.4 we know that $\psi(x,\theta) = \psi^{\text{lin}}(x,\theta,\theta^{(0)}) + \epsilon(x,\theta,\theta^{(0)})$ with $|\epsilon(x,\theta,\theta^{(0)}| \leq \frac{R^2}{2\sqrt{p}}$, for all x and all θ. Thus, the problem can be rewritten :

$$\left|\mathcal{L}^{\text{SATURNN}}(\theta) - \mathcal{L}^{\text{LR}}(\eta)\right| = \left|\frac{1}{N}\sum_{i=1}^{N} L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)},\theta,\theta^{(0)}) + \epsilon(x^{(i)},\theta,\theta^{(0)})\right),y^{(i)}\right) - L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)},\theta,\theta^{(0)})\right),y^{(i)}\right)\right|. \tag{57}$$

At first, we consider the problem for a fixed i. Two cases have to be considered according to the value of the true label:

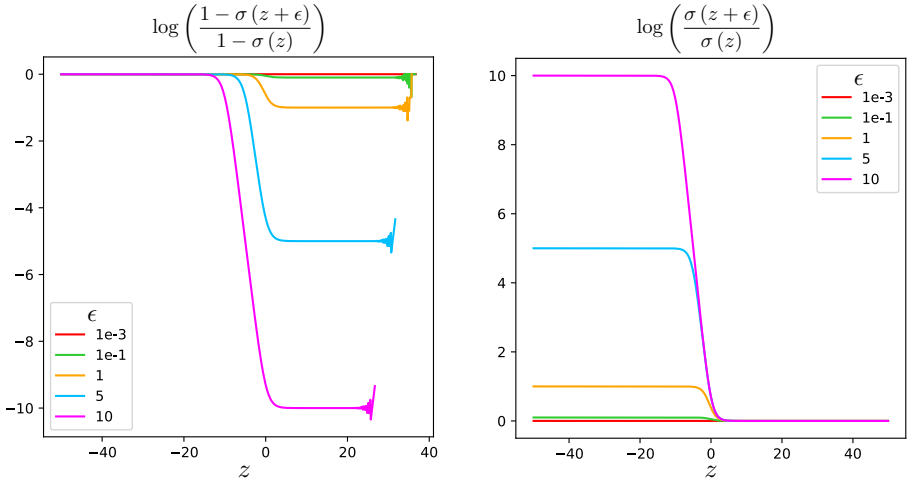

Figure 6. Plot of $|L(\sigma(z+\epsilon),y) - L(\sigma(z),y)|$ (58) for different values of ϵ according to the label: $y=0$ (left) and $y=1$ (right).

$$|L(\sigma(z+\epsilon),y) - L(\sigma(z),y)| = \begin{cases} \left|\log\left(\frac{1-\sigma(z+\epsilon)}{1-\sigma(z)}\right)\right| & \text{if } y=0, \\ \left|\log\left(\frac{\sigma(z+\epsilon)}{\sigma(z)}\right)\right| & \text{if } y=1. \end{cases} \tag{58}$$

Let consider the case $y = 0$ and $f_\epsilon(z) = \log\left(\frac{1-\sigma\ (z+\epsilon)}{1-\sigma(z)}\right)$ displayed in the Figure 6-left. We first compute the derivative of $f_\epsilon(z)$ and study its sign:

$$f_\epsilon^{'}(z) = \frac{1}{1+\exp(-z)} - \frac{1}{1+\exp(-z-\epsilon)}. \tag{59}$$

Let us assume that $\epsilon > 0$ (the case $\epsilon < 0$ will be considered later). We get $f_\epsilon^{'}(z) < 0$. Now that we have established that $f_\epsilon(z)$ is strictly decreasing (as confirmed by the Figure 6-left), we study its boundaries.

When z tends to $-\infty$, we get

$$\begin{aligned}
\lim_{z\to-\infty}\left|\log\left(\frac{1-\sigma\ (z+\epsilon)}{1-\sigma\,(z)}\right)\right| &= \lim_{z\to-\infty}\left|\log\left(\frac{1-\frac{1}{1+\exp(-z-\epsilon)}}{1-\frac{1}{1+\exp(-z)}}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(1-\frac{1}{1+\exp(-z-\epsilon)}\right) - \log\left(1-\frac{1}{1+\exp(-z)}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(\frac{\exp(-z-\epsilon)}{1+\exp(-z-\epsilon)}\right) - \log\left(\frac{\exp(-z)}{1+\exp(-z)}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(\frac{\frac{1}{\exp(z+\epsilon)}}{1+\frac{1}{\exp(z+\epsilon)}}\right) - \log\left(\frac{\frac{1}{\exp(z)}}{1+\frac{1}{\exp(z)}}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(\frac{\frac{1}{\exp(z+\epsilon)}}{\frac{1+\exp(z+\epsilon)}{\exp(z+\epsilon)}}\right) - \log\left(\frac{\frac{1}{\exp(z)}}{\frac{1+\exp(z)}{\exp(z)}}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(\frac{1}{1+\exp(z+\epsilon)}\right) - \log\left(\frac{1}{1+\exp(z)}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(\frac{1+\exp(z)}{1+\exp(z+\epsilon)}\right)\right| \\
&= \lim_{z\to-\infty}\left|\log\left(1+\exp(z)\right) - \log\left(1+\exp(z+\epsilon)\right)\right| \\
&= 0.
\end{aligned} \tag{60}$$

When z tends to $+\infty$, we get

$$\begin{aligned}
\lim_{z\to+\infty}\left|\log\left(\frac{1-\sigma\ (z+\epsilon)}{1-\sigma\,(z)}\right)\right| &= \lim_{z\to+\infty}\left|\log\left(\frac{1-\frac{1}{1+\exp(-z-\epsilon)}}{1-\frac{1}{1+\exp(-z)}}\right)\right| \\
&= \lim_{z\to+\infty}\left|\log\left(\frac{\frac{\exp(-z-\epsilon)}{1+\exp(-z-\epsilon)}}{\frac{\exp(-z)}{1+\exp(-z)}}\right)\right| \\
&= \lim_{z\to+\infty}\left|\log\left(\frac{\exp(-z-\epsilon)}{1+\exp(-z-\epsilon)} \times \frac{1+\exp(-z)}{\exp(-z)}\right)\right| \\
&= \lim_{z\to+\infty}\left|\log\left(\exp(-\epsilon) \times \frac{1+\exp(-z)}{1+\exp(-z-\epsilon)}\right)\right| \\
&= \lim_{z\to+\infty}\left|-\epsilon + \log\left(1+\exp(-z)\right) - \log\left(1+\exp(-z-\epsilon)\right)\right| \\
&= \epsilon.
\end{aligned} \tag{61}$$

Since $f_\epsilon(z)$ is strictly decreasing and admits lower and upper boundaries in (60) and (61), we can conclude that:

$$0 < |L(\sigma(z+\epsilon), y) - L(\sigma(z), y)| < \epsilon \leq \frac{R^2}{2\sqrt{p}}\ \forall z \in \mathbb{R} \text{ when } y = 0. \tag{62}$$

When $\epsilon < 0$, we get the same bound. The proof is straightforward but we must modify a little bit the calculation.

With exactly the same reasoning, we get

$$0 < |L(\sigma(z+\epsilon), y) - L(\sigma(z), y)| < \epsilon \leq \frac{R^2}{2\sqrt{p}} \; \forall z \in \mathbb{R} \; y = 1. \tag{63}$$

It follows that

$$\begin{aligned}
\sup_{\substack{\theta \in \mathcal{B}_2^{2p+1}(\theta^{(0)}, R) \\ \eta \in \mathcal{B}_2^{2p+1}(0,R)}} \left| \mathcal{L}^{\text{SATURNN}}(\theta) - \mathcal{L}^{\text{LR}}(\eta) \right| &= \sup_{\theta \in \mathcal{B}_2^{2p+1}(\theta^{(0)}, R)} \left| \frac{1}{N} \sum_{i=1}^{N} L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)}, \theta, \theta^{(0)}) + \epsilon(x, \theta, \theta^{(0)})\right), y^{(i)}\right) \right. \quad (64) \\
&\qquad \left. - L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)}, \theta, \theta^{(0)})\right), y^{(i)}\right) \right| \\
&= \frac{1}{N} \sum_{i=1}^{N} \sup_{\theta \in \mathcal{B}_2^{2p+1}(\theta^{(0)}, R)} \left| L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)}, \theta, \theta^{(0)}) + \epsilon(x^{(i)}, \theta, \theta^{(0)})\right), y^{(i)}\right) \right. \quad (65) \\
&\qquad \left. - L\left(\sigma\left(\psi^{\text{lin}}(x^{(i)}, \theta, \theta^{(0)})\right), y^{(i)}\right) \right| \\
&\leq \frac{1}{N} \sum_{i=1}^{N} \epsilon \\
&\leq \epsilon \\
&\leq \frac{R^2}{2\sqrt{p}} \quad (66) \\
&= O\left(\frac{1}{\sqrt{p}}\right). \quad (67)
\end{aligned}$$

□

F. Proof of Proposition 4.1

Proof. The kernel defined by (22) admits a finite expectation.

Let us recall that the features x lie on an open ball $\mathcal{B}_2^d(0,r)$ $r>0$ and the vector of initialized parameters is defined as $\theta^{(0)}=[\beta_0^{(0)},\beta_1^{(0)},\ldots,\beta_p^{(0)},b_1^{(0)},\ldots,b_p^{(0)}]$ with $\beta_k^{(0)}\sim\mathcal{N}(0,1)$ and $b_k^{(0)}\sim\mathcal{U}[-r,+r]$. We assume $s_k\in\{-1,1\}\sim\mathcal{B}(1/2)$ and $\upsilon(k)\sim\mathcal{U}[\![1,d]\!]$, for each $k\in\{1,\ldots,p\}$. Finally $\phi(\cdot)=\max\{0,\cdot\}$ is the ReLU activation and $\sigma(\cdot)$ the sigmoïd.

We compute the expectation of $\kappa(x,\tilde{x})$ defined by:

$$\kappa(x,\tilde{x})=\frac{1}{p}\left[1+\sum_{k=1}^{p}\phi\left(s_k x_{\upsilon(k)}+b_k^{(0)}\right)\phi\left(s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}\right)+{\beta_k^{(0)}}^2\mathbb{1}_{\{s_k x_{\upsilon(k)}+b_k^{(0)}>0\}}\mathbb{1}_{\{s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}>0\}}\right]. \tag{68}$$

We have :

$$\begin{aligned}
\mathbb{E}(\kappa(x,\tilde{x}))&=\mathbb{E}\left(\frac{1}{p}\left[1+\sum_{k=1}^{p}\phi\left(s_k x_{\upsilon(k)}+b_k^{(0)}\right)\phi\left(s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}\right)+{\beta_k^{(0)}}^2\mathbb{1}_{\{s_k x_{\upsilon(k)}+b_k^{(0)}>0\}}\mathbb{1}_{\{s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}>0\}}\right]\right)\\
&=\frac{1}{p}+\frac{1}{p}\sum_{k=1}^{p}\mathbb{E}\left(\phi\left(s_k x_{\upsilon(k)}+b_k^{(0)}\right)\phi\left(s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}\right)+{\beta_k^{(0)}}^2\mathbb{1}_{\{s_k x_{\upsilon(k)}+b_k^{(0)}>0\}}\mathbb{1}_{\{s_k\tilde{x}_{\upsilon(k)}+b_k^{(0)}>0\}}\right)\\
&=\frac{1}{p}+\frac{1}{p}\sum_{k=1}^{p}\mathbb{P}(s_k=-1)\mathbb{E}\left(\phi\left(-x_{\upsilon(k)}+b_k^{(0)}\right)\phi\left(-\tilde{x}_{\upsilon(k)}+b_k^{(0)}\right)\right)\\
&\qquad+\mathbb{P}(s_k=1)\mathbb{E}\left(\phi\left(x_{\upsilon(k)}+b_k^{(0)}\right)\phi\left(\tilde{x}_{\upsilon(k)}+b_k^{(0)}\right)\right)\\
&\qquad+\mathbb{P}(s_k=-1)\mathbb{E}\left({\beta_k^{(0)}}^2\right)\mathbb{E}\left(\mathbb{1}_{\{-x_{\upsilon(k)}+b_k^{(0)}>0\}}\mathbb{1}_{\{-\tilde{x}_{\upsilon(k)}+b_k^{(0)}>0\}}\right)\\
&\qquad+\mathbb{P}(s_k=1)\mathbb{E}\left({\beta_k^{(0)}}^2\right)\mathbb{E}\left(\mathbb{1}_{\{x_{\upsilon(k)}+b_k^{(0)}>0\}}\mathbb{1}_{\{\tilde{x}_{\upsilon(k)}+b_k^{(0)}>0\}}\right),
\end{aligned}\tag{69}$$

with $\mathbb{P}(s_k=-1)=\mathbb{P}(s_k=1)=\frac{1}{2}$ as $s_k\in\{-1,1\}\sim\mathcal{B}(1/2)$, $\mathbb{E}\left({\beta_k^{(0)}}^2\right)=1$ since $\beta_k^{(0)}\sim\mathcal{N}(0,1)$ and $\upsilon(k)\sim\mathcal{U}[\![1,d]\!]$, for each $k\in\{1,\ldots,p\}$. Thus we have:

$$\begin{aligned}
\mathbb{E}(\kappa(x,\tilde{x}))&=\frac{1}{p}+\frac{1}{2p}\sum_{k=1}^{p}\sum_{i=1}^{d}\mathbb{P}(\upsilon(k)=i)\mathbb{E}\left(\phi\left(-x_i+b_k^{(0)}\right)\phi\left(-\tilde{x}_i+b_k^{(0)}\right)\right)\\
&\qquad+\mathbb{P}(\upsilon(k)=i)\mathbb{E}\left(\phi\left(x_i+b_k^{(0)}\right)\phi\left(\tilde{x}_i+b_k^{(0)}\right)\right)\\
&\qquad+\mathbb{P}(\upsilon(k)=i)\mathbb{E}\left(\mathbb{1}_{\{-x_i+b_k^{(0)}>0\}}\mathbb{1}_{\{-\tilde{x}_i+b_k^{(0)}>0\}}\right)\\
&\qquad+\mathbb{P}(\upsilon(k)=i)\mathbb{E}\left(\mathbb{1}_{\{x_i+b_k^{(0)}>0\}}\mathbb{1}_{\{\tilde{x}_i+b_k^{(0)}>0\}}\right)\\
&=\frac{1}{p}+\frac{1}{2pd}\sum_{k=1}^{p}\sum_{i=1}^{d}\mathbb{E}\left(\left[-x_i+b_k^{(0)}\right]\left[-\tilde{x}_i+b_k^{(0)}\right]\mathbb{1}_{\{b_k^{(0)}>x_i,b_k^{(0)}>\tilde{x}_i\}}\right)\\
&\qquad+\mathbb{E}\left(\left[x_i+b_k^{(0)}\right]\left[\tilde{x}_i+b_k^{(0)}\right]\mathbb{1}_{\{b_k^{(0)}>-x_i,b_k^{(0)}>-\tilde{x}_i\}}\right)\\
&\qquad+\mathbb{E}\left(\mathbb{1}_{\{b_k^{(0)}>x_i\}}\mathbb{1}_{\{b_k^{(0)}>\tilde{x}_i\}}\right)+\mathbb{E}\left(\mathbb{1}_{\{b_k^{(0)}>-x_i\}}\mathbb{1}_{\{b_k^{(0)}>-\tilde{x}_i\}}\right).
\end{aligned}$$

Let $\mathrm{f}_b(t)$ denote the probability density function of b. Since we suppose $b_k^{(0)}\sim\mathcal{U}[-r,+r]$ for each $k\in\{1,\ldots,p\}$, we have $\mathrm{f}_b(t)=\frac{1}{2r}$.

$$
\begin{aligned}
\mathbb{E}(\kappa(x,\tilde{x})) &= \frac{1}{p} + \frac{1}{2pd}\sum_{k=1}^{p}\sum_{i=1}^{d}\int_{-r}^{r}[-x_i+t][-\tilde{x}_i+t]\mathrm{f}_b(t)\mathbb{1}_{\{t>x_i,t>\tilde{x}_i\}}\,\mathrm{d}t + \int_{-r}^{r}[x_i+t][\tilde{x}_i+t]\mathrm{f}_b(t)\mathbb{1}_{\{t>-x_i,t>-\tilde{x}_i\}}\,\mathrm{d}t \\
&\qquad + \int_{-r}^{r}\mathrm{f}_b(t)\mathbb{1}_{\{t>x_i,t>\tilde{x}_i\}}\,\mathrm{d}t + \int_{-r}^{r}\mathrm{f}_b(t)\mathbb{1}_{\{t>-x_i,t>-\tilde{x}_i\}}\,\mathrm{d}t \\
&= \frac{1}{p} + \frac{1}{2pd}\sum_{k=1}^{p}\sum_{i=1}^{d}\int_{\max(x_i,\tilde{x}_i)}^{r}\frac{1}{2r}[-x_i+t][-\tilde{x}_i+t]\,\mathrm{d}t + \int_{\max(-x_i,-\tilde{x}_i)}^{r}\frac{1}{2r}[x_i+t][\tilde{x}_i+t]\,\mathrm{d}t \\
&\qquad + \int_{\max(x_i,\tilde{x}_i)}^{r}\frac{1}{2r}\,\mathrm{d}t + \int_{\max(-x_i,-\tilde{x}_i)}^{r}\frac{1}{2r}\,\mathrm{d}t \\
&= \frac{1}{p} + \frac{1}{2pd}\sum_{k=1}^{p}\sum_{i=1}^{d}\frac{1}{2r}\left[\left[x_i\tilde{x}_i t - \frac{t^2}{2}(x_i+\tilde{x}_i) + \frac{t^3}{3}\right]_{\max(x_i,\tilde{x}_i)}^{r} + [t]_{\max(x_i,\tilde{x}_i)}^{r}\right. \\
&\qquad \left. + \left[x_i\tilde{x}_i t + \frac{t^2}{2}(x_i+\tilde{x}_i) + \frac{t^3}{3}\right]_{\max(-x_i,-\tilde{x}_i)}^{r} + [t]_{\max(-x_i,-\tilde{x}_i)}^{r}\right] \\
&= \frac{1}{p} + \frac{1}{4rd}\sum_{i=1}^{d}\left[2r(x_i\tilde{x}_i+1) + \frac{2r^3}{3} + \min(x_i,\tilde{x}_i) - \max(x_i,\tilde{x}_i) - x_i\tilde{x}_i\left(\max(x_i,\tilde{x}_i) - \min(x_i,\tilde{x}_i)\right)\right. \\
&\qquad \left. + \frac{1}{2}(x_i+\tilde{x}_i)\left(\max(x_i,\tilde{x}_i)^2 - \min(x_i,\tilde{x}_i)^2\right) - \frac{1}{3}\left(\max(x_i,\tilde{x}_i)^3 - \min(x_i,\tilde{x}_i)^3\right)\right] \\
&= \frac{1}{p} + \frac{r^2}{6} + \frac{1}{4rd}\sum_{i=1}^{d}\left[2r(x_i\tilde{x}_i+1) + \min(x_i,\tilde{x}_i) - \max(x_i,\tilde{x}_i) - x_i\tilde{x}_i(\max(x_i,\tilde{x}_i) - \min(x_i,\tilde{x}_i)\right. \\
&\qquad \left. + \frac{1}{2}(x_i+\tilde{x}_i)(\max(x_i,\tilde{x}_i)^2 - \min(x_i,\tilde{x}_i)^2) - \frac{1}{3}(\max(x_i,\tilde{x}_i)^3 - \min(x_i,\tilde{x}_i)^3)\right]. \quad (70)
\end{aligned}
$$

Since $\max(a,b) = \frac{a+b+|a-b|}{2}$ and $\min(a,b) = \frac{a+b-|a-b|}{2}$ for $a, b \in \mathbb{R}^2$, we finally have:

$$
\mathbb{E}(\kappa(x,\tilde{x})) = \frac{1}{p} + \frac{r^2}{6} + \frac{1}{4rd}\sum_{i=1}^{d} 2r(x_i\tilde{x}_i+1) - |x_i-\tilde{x}_i| + \frac{1}{6}|x_i-\tilde{x}_i|^3. \quad (71)
$$

Moreover, a simple calculation, as it is done for the expectation, shows that $\mathbb{V}(\kappa_0(x,\tilde{x})) = \frac{1}{p^2}[f(x,\tilde{x},\theta,\theta^{(0)})]$, where $f(x,\tilde{x},\theta,\theta^{(0)})$ is a continuous function that does not depend on p. Hence, it follows that

$$
\mathbb{V}(\kappa_0(x,\tilde{x})) = O\left(\frac{1}{p^2}\right). \quad (72)
$$

□

G. Supplementary Experimental Results

G.1. Estimated decision rules

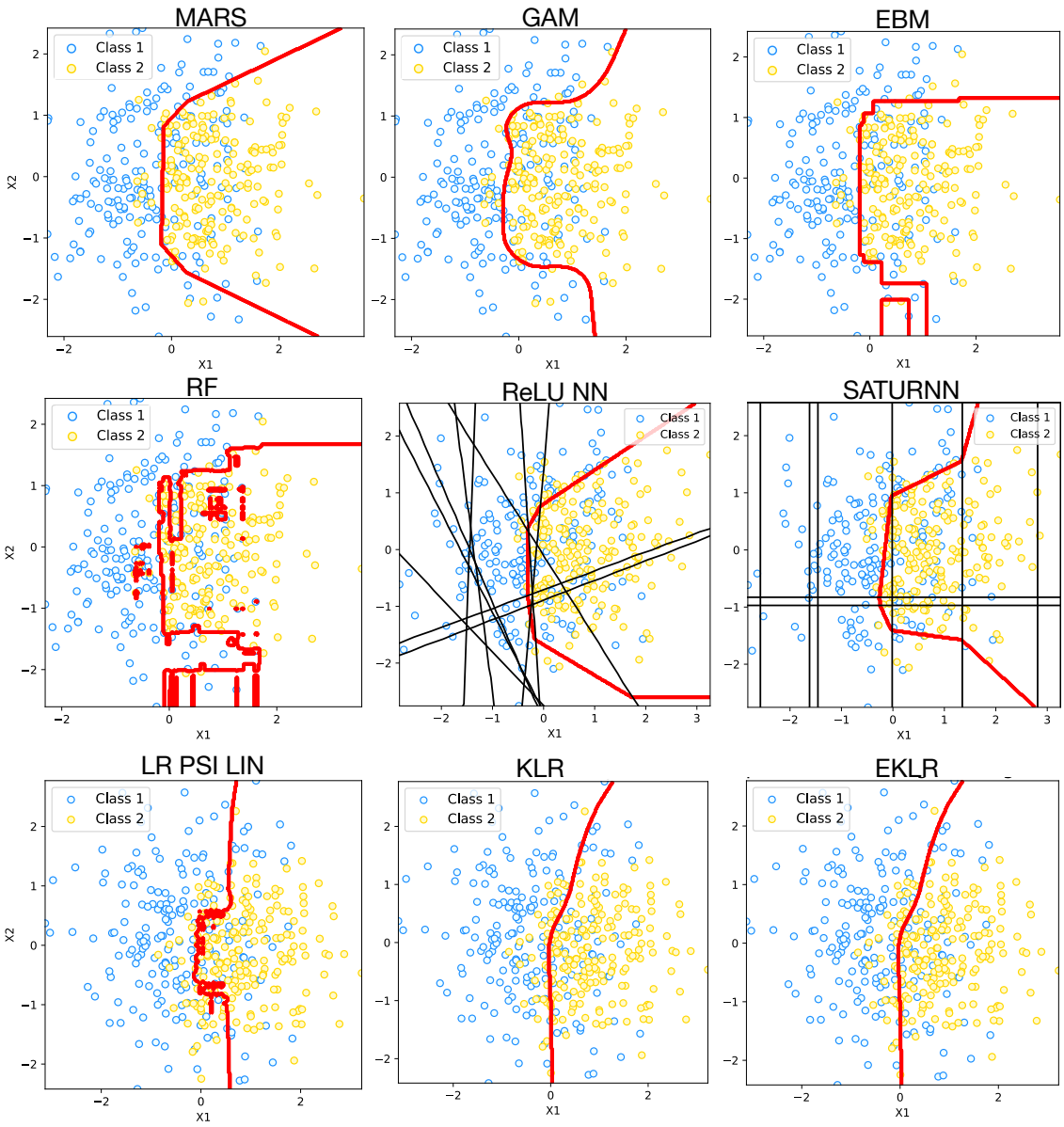

Figure 7. Gaussian Dataset: Estimated decision boundaries in red for MARS (Friedman, 1991), GAM (Hastie, 2017), EBM (Xie et al., 2020), RF (Breiman, 2001), ReLU NN ($p = 10$) (1), SATURNN ($p = 10$) (6), LR on locally linearized score function ($p = 30000$) (15), KLR ($p = 30000$) (23) and EKLR (32) ($p = 30000$). Table 1 reports their respective accuracies.

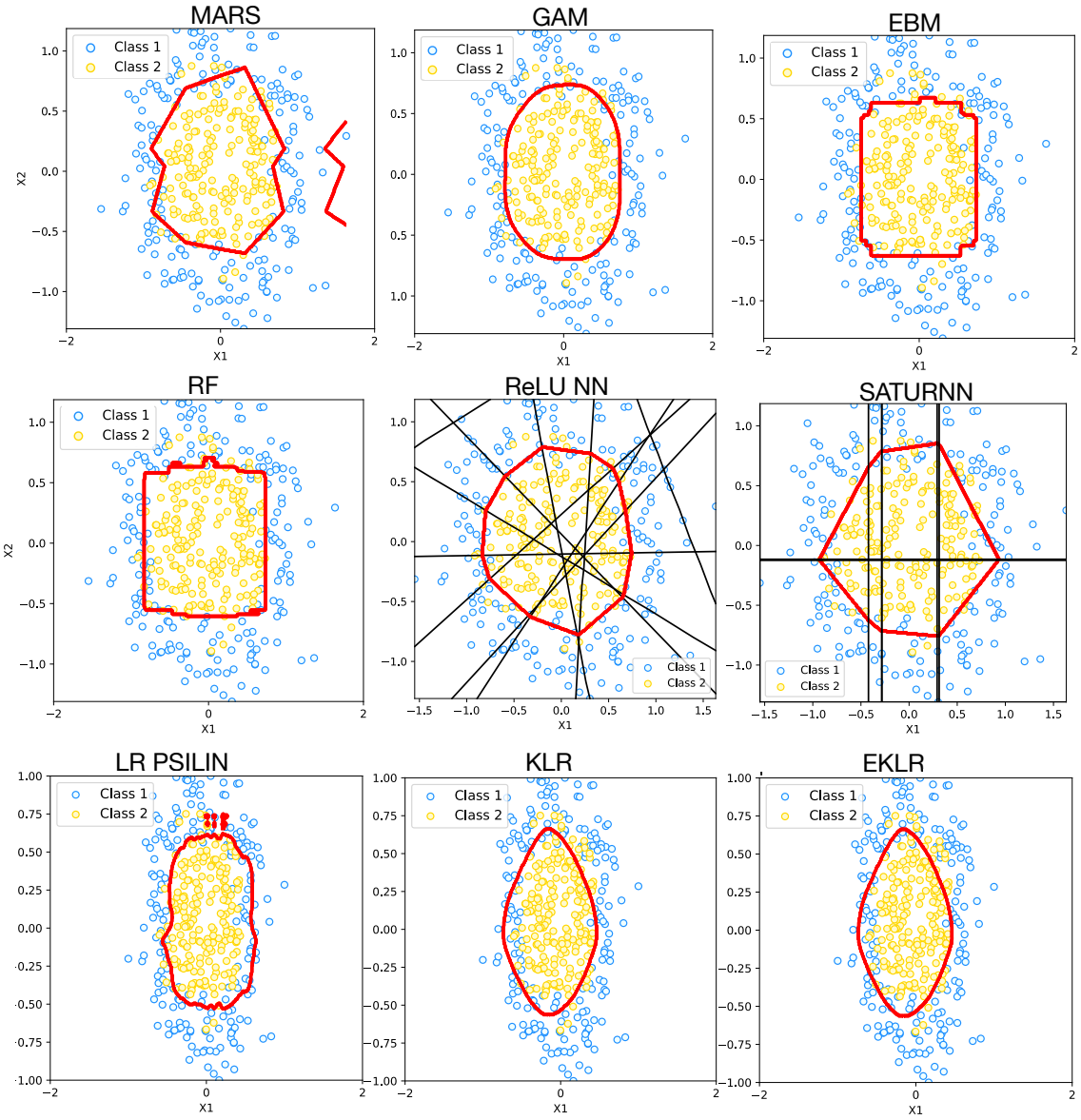

Figure 8. Circle Dataset: Estimated decision boundaries in red for MARS (Friedman, 1991), GAM (Hastie, 2017), EBM (Xie et al., 2020), RF (Breiman, 2001), ReLU NN ($p = 10$)(1), SATURNN ($p = 10$) (6), LR on locally linearized score function ($p = 50000$) (15), KLR ($p = 50000$) (23) and EKLR ($p = 50000$) (32). Table 1 reports their respective accuracies.

G.2. Estimated splines on Circle Dataset

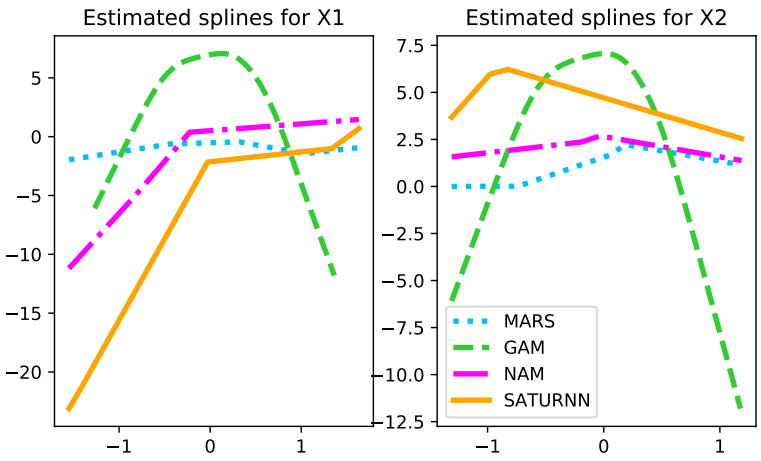

Figure 9. Circle Dataset: estimated splines for $X1$ (left) and $X2$ (right).

G.3. Illustration of Theorem 3.4 on Circle Dataset

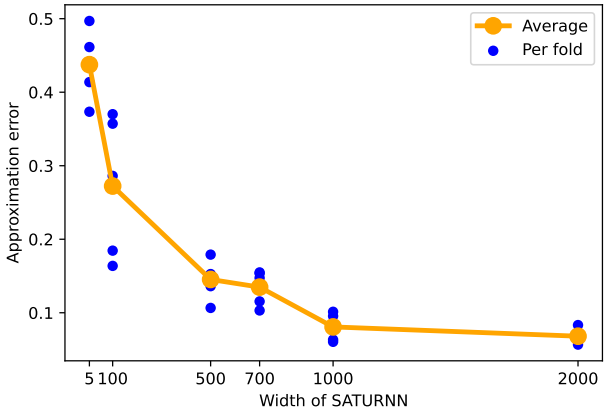

Figure 10. Circle Dataset: mean of the approximation errors of SATURNN by $\partial_{\mathrm{LR}}(x, \eta)$ (15) with $\hat{\theta}^{\mathrm{SATURNN}}$ in (6) over 5 folds for different values of p.