Mathematical details of
Sensitivity analysis and Explainable AI

M.Sc. (TUM) Aydin Javadov

Contents

1 Theoretical Background

In this section we first go over the fundamentals on explainable artificial intelligence. We then proceed with a review of sensitivity analysis methods and the discussion of different sampling techniques. Finally, for potential representation learning frameworks, we examine the existing literature on explainable artificial intelligence (XAI) methodologies for the clustering process. By following this structure, we provide a comprehensive overview of the research landscape and establish a strong mathematical foundation for the conjunction of sensitivity analysis, explainable ai and representation learning.

1.1 Fundamentals of XAI

1.1.1 Defining Explainability vs. Interpretability

It is essential to develop a precise understanding of the ideas of interpretability and explainability in order to lay a strong foundation. Often, researchers use these words interchangeably. Although no clear separation is defined with mathematical rigor, notable efforts have been made to distinguish these two concepts [29] [2]. A popular definition of interpretability by Doshi Velez & Kim states that "it is the ability to explain or present in humanly understandable terms" [29] [27]. In his definition, Miller describes interpretability as "the degree to which a human can understand the cause of a decision" [33]. Linardatos et al., in their overview of XAI, after going over major definitions, conclude that interpretability is mostly about the intuition behind the outputs of the model, such that the better the interpretability, the easier it is to identify the cause-effect relationships between the input and outputs [29]. According to Linardatos et al. and Doshi Velez & Kim, explainability is a tighter concept, and it is linked to internal mechanisms of the ML system [29][12]. Thus, making a model explainable means making the algorithm's inner processes and underlying logic more transparent to humans. Inspired by these, Gilpin et al. concluded that the *explainability* of the model is critical for developing transparent methods that can be used in the presence of ethical concerns [16]. Hence, we focus on the term explainability throughout the work.

1.1.2 Importance

Consider the question: "Why don’t we just accept the model and ignore its reasoning if a machine learning model performs well?". Doshi-Velez et al. answer: "The issue is that most real-world tasks cannot be fully described by a single statistic, such as classification accuracy." [12][34] Digging further into the importance of explainability, we can see that there is a trade-off to be made when using predictive modeling: Do we simply want to be informed of the predictions? Alternatively, do we want to understand the reasoning behind the prediction and risk losing (not even always) some predictive accuracy in exchange? There might be occasions where it is sufficient to know whether the predictive performance of the model is good enough without caring much about why a decision was made. This may happen because of the use cases of a model generally comprised of low-risk situations, where a mistake will not have serious repercussions (like a movie recommender system) [34]. One other reason might be that the technique has already been thoroughly researched and assessed (like optical character recognition), and no further explainability power is desired. However, there might be other circumstances, presumably critical in nature, where understanding the "why" can aid in our understanding of the issue, the data, and the possible causes of a model’s failure [34]. The critical point that needs to be highlighted is that debugging machine learning models is most effective and efficient when the models have been *explained* to us.

1.1.3 Taxonomy

There are several essential notions as per the potential approaches to model explainability. As mentioned before, the innate nature of XAI is to try to explain AI models, which are usually classified [34] as *white-box*, sometimes called glass-box, where it is possible to intuitively reason about the inner processes, and *black-box* approaches, whose name speaks for itself—little to no clue for an explanation. This work primarily focuses on the latter version.

Moreover, XAI methods can be *Post-hoc* or *Intrinsic*. The classification of machine learning approaches as post-hoc or intrinsic is determined based on how they address explainability. Intrinsic approaches involve constraining the complexity of the machine learning model during training, while post-hoc methods analyze an already trained model to achieve explainability [34]. Short decision trees and sparse linear models are examples of machine learning models that are intrinsically explainable because of their straightforward form. On the other hand, post-hoc methods take the trained model as is and employ additional techniques to achieve explainability. An example of a post-hoc interpretation method is the importance of permutation features [34].

Another differentiation between XAI methods is being *model-specific* or *model-agnostic*. Model-specific interpretation tools are only available for a subset of model classes. As an elementary example, since the interpretation of inherently interpretable (explainable) models is always model-specific, the interpretation (explanation) of the weights of a linear model is model-specific. Model-specific methods are those that are only effective for the explanation of, say, neural networks. An important note is that model-agnostic methods are employed post-hoc (after the model has been trained) and can be used in any machine-learning model of the given context [34]. These agnostic techniques typically operate by examining feature input and output pairs. By definition, these techniques cannot access model internals like weights or structural data.

Explaining the model can be projected to the concept of sensitivity analysis in certain contexts. The next part briefly discusses the motivation for relying our study on sensitivity analysis.

1.2 Sensitivity Analysis & Explainable AI

Numerous descriptions of sensitivity analysis agree on a common definition; sensitivity analysis is the study of how different sources of uncertainty in an input can be separated and assigned to the output of a mathematical model or system [46] [48]. As Stein et al. mention in their technical reviews, sensitivity analysis techniques are essential for better comprehending the impact and uncertainty of features or parameters in machine learning models, simulators, and practical implementations [61]. Sensitivity analysis can be seen as a model-agnostic approach in the context of explainable AI (XAI) because it can provide extensive insights about machine learning models and applications without any specific information about the model. In fact, sensitivity analysis plays an important role in XAI as it can answer some of these questions without being dependent on a certain machine learning model or even a sampling strategy [61]. Especially when the number of model inputs is large, recognizing the factors on which to focus resources in data collection and data-driven modeling efforts becomes crucial. The literature contains a number of XAI techniques. Saliency Maps [55], adversarial examples [63], GradCAM [53], and other model-specific XAI techniques are a few examples. The core ideas of Shap [31], LIME [44], and some other methods, which switch sets of inputs on and off to determine which traits contribute most to a particular prediction, are similar to those of the far more well-known GSA methods[61]. Sensitivity analysis is used in many real-world applications, including but not limited to understanding the engine models [45][19], understanding the vehicle dynamics [71], and understanding the lifecycle of combustion engine cars [17].

1.3 Sensitiviy Analysis

1.3.1 Basic Definitions and concepts

Aims of Sensitivity Analysis

Understanding what Sensitivity Analysis is capable of achieving is crucial before moving further. The following objectives are widely identified in the literature:

- Ranking:

  Sorting the input features according to how they affect the variability of model output. The higher-ranked features are, therefore, the focus of experimental or numerical estimation. Such ranking can be rooted in several formulations (which will be discussed in upcoming parts), for example, because the output uncertainty (variance) is reduced the most when these input values are removed (fixed).

- Screening:

  Screening determines which model inputs are indistinguishable, i.e., those with little to no impact on the variability of mode output. One use case would be, for instance, the dimensionality reduction of the problem by putting these inputs at fixed values.

Global vs. Local Sensitivity Analysis

Based on the input space exploration depth, Sensitivity Analysis approaches can be divided into different categories. Local techniques concentrate on analyzing the sensitivity of model inputs at a particular location. Global techniques, on the other hand, collect sensitivities at numerous places in the input space before calculating some measure of the average of these sensitivities. This averaged value then shows the input's influence on the output's uncertainty [43]. Local techniques are extensively used because they are simple and computationally efficient, yet, they are most beneficial for linear models. Nonlinear models may give incorrect conclusions when extrapolating sensitivity [46]. Global techniques, on the other hand, are effective for nonlinear models as they can expose interactions between inputs.

One-at-a-time and Many-at-a-time SA

Sensitivity analysis techniques imply model (re)evaluations using various sets of input data. One way to categorize these techniques is to simply count the number of input values that change for each subsequent model simulation.

- One-at-a-time (OAT) methods: Altering one feature per run
- Many-at-a-time (MAT) methods: Altering multiple features per run.

The majority of one-at-a-time algorithms begin with a basic set of input values where it is known that the model will converge [43]. Hence, altering the value of one input feature lowers the likelihood that the model evaluation will be unsuccessful for any reason, like instability or numerical mistake. Beyond their resistance to convergence problems, another benefit of OAT methods is that if the model evaluation of a set of baseline values differs from that when one of the inputs is changed, we can attribute the cause of the difference to that particular input because it must have some impact on the output, at least at that particular location in input space. However, OAT approaches have some significant drawbacks, most notably a challenge in the analysis of nonlinear models [49][43]. While local methods can only be OAT-based, global sensitivity techniques can be OAT or MAT-based. Although MAT is computationally more expensive and more likely to evaluate the model at input values where it is unstable, they are recommended for the analysis of nonlinear functions since they cover a larger amount of the factor space [46][49][43].

1.3.2 Review and Mathematical Formulation of relevant Sensitivity Analysis Methods

Variance based Sensitivity Analysis

The goal of the variance-based sensitivity analysis methods is to estimate the portion of model variance caused by each input feature and how that feature interact with every other input feature. In the following two sections, we mention two of such approaches, which, with proper adjustment and tweaks, are used in our methodologies.

Sobol Sensitivity Analysis

We start by mentioning Sobol method [58][59], first proposed by Ilya Sobol in 1993. It is a variance decomposition method, such that the variance of the model's output is being decomposed into summands of variances of the input feature. Thus, Sobol Sensitivity Analysis (SSA) allows one to observe the contribution of each input feature for the generation of output variance. The sensitivity indices induced by Sobol method can be represented via conditional probability formulations [22][36][50]. The direct involvement of each input feature to the output variance is called its *main effect* on output. Main effects of features are also alternatively called *first-order sensitivity indices* of input features. Equation 1.1 demonstrates the calculation of such first-order indices:

$$S_i = \frac{Var_{X_i}(E_{X_{\sim i}}(Y|X_i))}{Var(Y)} \tag{1.1}$$

Here X_i means i_{th} input feature, whereas $X_{\sim_i}$ means everything but the i_{th} input feature. $E(\cdot)$ and $Var(\cdot)$ represent expected value and variance, respectively. For the expected value operator, the mean of Y is taken over all possible values of $X_{\sim_i}$ (which means keeping the i_{th} feature fixed). Then the variance of these expected values is computed over all possible values of for X_i. One way to interpret the numerator ($Var_{X_i}(E_{X_{\sim i}}(Y|X_i))$) is to consider it as the expected reduction in model variance that would be obtained if X_i was to be fixed [43]. Moreover, it is quite usual for the systems to have interactions among the input features (e.g. input X_2 is especially important when X_5 exhibits certain behaviors), which might also contribute to variance generation in the model output. These total contributions are called *total-sensitivity indices* of the inputs [18], and computed as in Equation 1.2:

$$S_{T_i} = \frac{E_{X_{\sim i}}(Var_{X_i}(Y|X_i))}{Var(Y)} \tag{1.2}$$

Here numerator ($E_{X_{\sim i}}(Var_{X_i}(Y|X_i))$) is the expected variance that would remain after fixing every variable except X_i. Thus, if the total sensitivity index for k_{th} feature is computed to be 0, this means the model's output shows no variation to k_{th} feature - feature k is then noninfluential.
We now proceed with more rigorous representation of Sobol SA.

Rigorous representation of Sobol Indices

Let $z = (z_1, z_2, \ldots, z_{n-1}, z_n)$ be the input features of a model. Considering that each feature's value has its own interval to vary, to make things simpler, we can normalize the intervals for all features and make them all live in $[0, 1]$. For simplicity, then assume that all features are distributed uniformly in [0,1], whereas the features are independent of one another. Call the model output a function of z, $f(z)$. In the context of probabilistic interpretation of the parameters, $f(z)$ is a random variable, with a mean f_0 (Equation 1.3) and variance D (Equation 1.4).

$$f_0 = \int f(z)dz \tag{1.3}$$

$$D = \int f(z)^2 dz - f_0^2 \tag{1.4}$$

In above equations all integrals are multiple integrals with limits $[0, 1]$ for every dimension, as per our previous definitions. The Sobol method relies on breaking down

1.4 into individual contributions from single features, as well as combined effects from pairs, from triples and so on. To do so, first we break down $f(z)$ as:

$$f(z) = f_0 + \sum_{i=1}^{n} f_i(z_i) + \sum_{i=1}^{n} \sum_{i \neq j}^{s} f_{ij}(z_i, z_j) + \cdots + f_{1\ldots n}(z_1, z_2, \ldots, z_n) \tag{1.5}$$

The components of the breakdown are formulated as:

$$f_{ij}(z_i, z_j) = \int f(z) \prod_{k \neq i,j} dz_k - f_0 \tag{1.6}$$

$$\int f_{i1,\ldots,in}(z_{i1}, \ldots, z_{in}) dz_k = 0 \tag{1.7}$$

and so on.
Next, we obtain ANOVA (Analysis of Variance) representation for the model output function, $f(z)$, which is based on the condition given as:

$$\int f_{i1,\ldots,in}(z_{i1}, \ldots, z_{in})\, dz_k = 0 \quad \text{for } k = i_1, \ldots, i_n. \tag{1.8}$$

As a result of this property, squaring both sides of 1.5 and integrating yields:

$$D = \sum_{i=1}^{k} D_i + \sum_{i<j} D_{ij} + \sum_{i<j<l} D_{ijl} + \cdots + D_{1,2,\ldots,k} \tag{1.9}$$

where $D_{1,\ldots,n} = \int f^2_{1,\ldots,n}(z_1, z_2 \ldots z_n) dz_1 \ldots dz_n$ is the variance of $f_{1,\ldots,n}(z_1, z_2 \ldots z_n)$, the partial variance corresponding to the given subset of features. Finally, the Sobol Indices for the given subset of features are computed as:

$$S_{1,\ldots,n} = \frac{D_{1,\ldots,n}}{D} \tag{1.10}$$

Mapping to the previous formulation in 1.1, First-Order Sobol index of i_{th} feature is given as:

$$S_i = \frac{D_i}{D} \tag{1.11}$$

This formulation can be extended to Second Order Index of ith and j_{th} features, representing the the contribution of the interaction between them: $S_{ij} = \frac{D_{ij}}{D}$. Eventually, one can figure out the Total order index for the i_{th} feature, representing the overall contribution of this feature including its main effect and all interactions with other features. Calculation of Total Order Indices (equivalent to 1.2) is given as:

$$S_{T_i} = S_i + \sum_{j \neq i} S_{ij} + \cdots + S_{1\ldots n} \tag{1.12}$$

It is worthwhile to note that, thanks to such formulation of the so called total sensitivity index, Sobol method can escape from the curse of dimensionality [47].

Key Steps for Conducting Sobol Sensitivity Analysis

The main guideline for performing Sobol analysis consists of preparing parameter sets and their ranges, sample generation, model simulations, the computation of Sobol indices based on the simulation results, and the analysis of the results. Depending on the outcomes and evaluations, we rollback to the pre-sobol step to redefine the setup and start over. See Figure 1.1 for the diagram of the workflow visualized. For the step of parameter set generation, we use the notion of Sobol Sequences and their Saltelli extension. We mention these and more sampling strategies in greater detail in 1.4.

We can now sum up the characteristics of the Sobol method. To begin with, no assumptions are being made between the inputs and outputs; instead, sole observation of the model behavior is being reported. Additionally, the reported indices comprise all the input features and all possible interactions, given the individual ranges. However, there are some handicaps as well. Like the other variance-based methods, the algorithm's biggest challenge is the high computational demand [43]. Another critical aspect of the algorithm is that the input features of the model should be independent to achieve the desired results effectively [4][70][43][59]. For this reason, it is essential to check for the potential high correlations of the input features.

Random Balanced Design Fourier Amplitude Sensitivity Testing (RBD-FAST)

In fact, several numerical techniques have been proposed in the literature ([59]; [47]; [62]; [46]) to evaluate the variance-based sensitivity indices of specific scalar inputs. The *frequency-based* approach, RBD-FAST, in particular offers a reliable and accurate estimation of all first-order sensitivity indices (main effects) with just one sample set of N simulations. In this section, we outline a technique for quickly, inexpensively, and precisely determining the RBD-FAST sensitivity indices, ψ_i. RBD-FAST integrates two concepts: the Fourier Amplitude Sensitivity Test (FAST) ([47]) and random balance experimental designs [51].

Fourier Amplitude Sensitivity Test (FAST)

We first start with the formulation of the classical baseline algorithm, FAST. Consider a black-box model $f(\hat{x})$, outputting y, where $\hat{x}$ is a input vector of d features: $(x_1, \dots, x_d)$. Without loss of generality, assume that the domain input features is the unit hypercube:

$$C^d = \{x \,|\, 0 \leq x_i \leq 1; i = 1, \dots, d\} \tag{1.13}$$

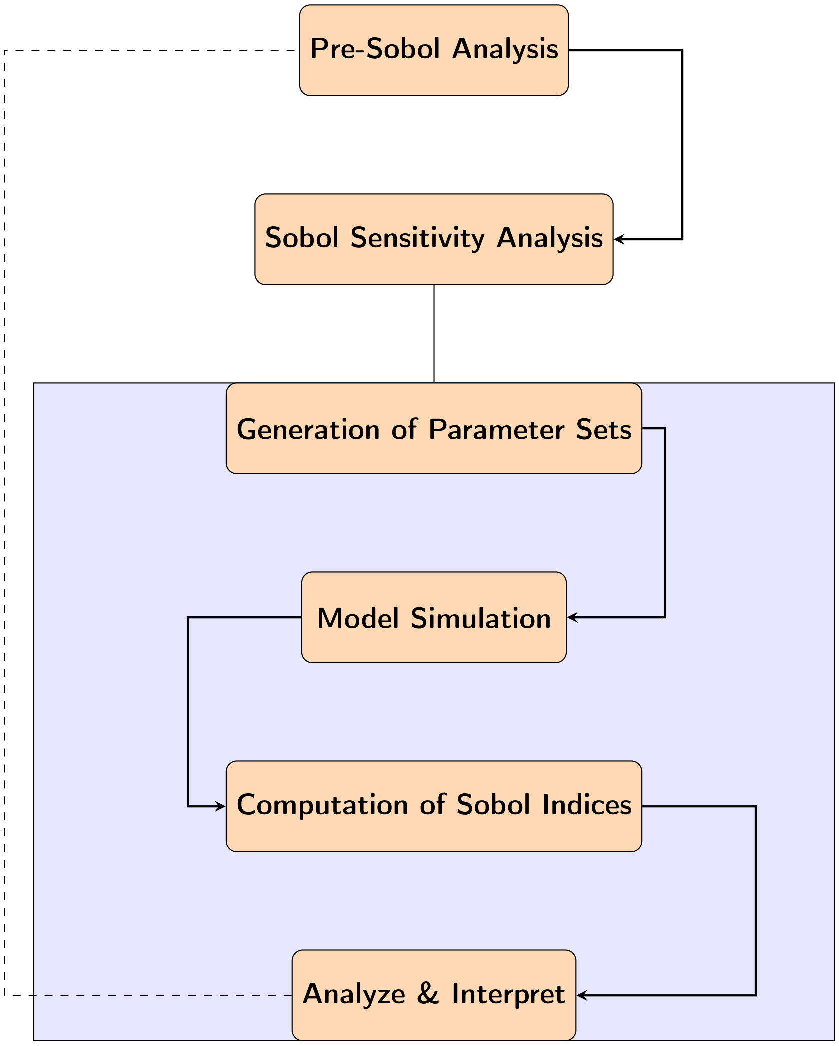

Pre-Sobol Analysis
Sobol Sensitivity Analysis
Generation of Parameter Sets
Model Simulation
Computation of Sobol Indices
Analyze & Interpret

Figure 1.1: General workflow of Sobol SA

We can assume that $\hat{x}$ is a random vector, with a certain probability density function, $P(\hat{x})$. Note that the p_{th} moment of the output y is given as:

$$\langle y^{(p)} \rangle = \int_{C^d} f^p(x_1, \ldots, x_d) P(x_1, \ldots, x_d)\, dx \tag{1.14}$$

The very first proposers of FAST, Cukier et al. [9], had proved that via the utilization of the multidimensional Fourier transformation (multiFt) of f, it is theoretically possible to obtain a variance decomposition (similar to ANOVA) of the variance of y in terms of the input features and their interactions. The problem, however, is a very high computational complexity led by the multiFt. Hence, authors had shifted to compute monodimensional Fourier transformation (monoFt) instead. Such monoFt is done along a certain curve and exploring the space of the unit hypercube, C^d. This curve is characterized by a set of parametric equations:

$$x_i = T_i(\sin \omega_i s), \quad \forall i = 1, \ldots, d \tag{1.15}$$

Here s is a scalar variable, and varies as $-\infty < s < \infty$. $w_i, \forall i = 1, \ldots, d$ comprises a set of different angular frequencies associated with each input feature. T_i are transformation functions for which several options were proposed throughout the literature. [9] originally suggested:

$$T_i : x_i \mapsto x_i e^{v_i \sin \omega_i s}, \quad \forall i = 1, \ldots, d \tag{1.16}$$

Here v_i represents the endpoints of the assumed (estimated) ranges of the variation of the feature x_i. Furthermore s lives in $(\frac{-\pi}{2}, \frac{\pi}{2})$. In their review of the FAST method, Saltelli et al. [47] had shown that such transformation 1.16 fits merely for features possessing long-tailed and positively skewed probability density functions. Another type of transformation function was proposed by Koda et al. [23]:

$$T_i : x_i \mapsto x_i(1 + v_i \sin(\omega_i s)) \tag{1.17}$$

Again in the review by [47], Equation 1.17 is shown to possess an U-shaped probability density function (thick tails but slim middle region). Hence, such transformation function does not fit for the uniformly distributed features, which would rather be a more general setup to begin with. Concurring with that, [9] suggests a general differential equation form, whose solution leads to the optimal search curve:

$$\pi \sqrt{(1 - x_i^2} P_i(T_i) \frac{dT_i(x_i)}{dx_i} = 1 \tag{1.18}$$

Here P_i is the assumed probability density function of the feature x_i. The transformation function which is the solution to this differential equation is proposed by [47], and is given as:

$$T_i : x_i \mapsto \frac{1}{2} + \frac{1}{\pi}\arcsin(\sin \omega_i s) \tag{1.19}$$

Such transformation function generates a set of oscillating straight lines in $[0, 1]$, whose empirical distribution can be considered as uniform [47].

Now, we continue from where we left off at Equation 1.15 ($x_i = T_i(\sin \omega_i s)$). The core idea here is that whatever the f and T_i are, the variation of s in $[-\infty, +\infty]$ results in a systematical search over C^d, simultenously over all the input features. During this process, each x_i oscillates with the respective ω_i, thus y exhibits various periodicities along with various frequencies ω_i. If x_i significantly affects y, then there will be considerable amplitude oscillations of y at the frequency w_i. This serves as the foundation for calculating a sensitivity measure that is based on the coefficients of the appropriate frequency w_i and its harmonics. We say that the curve we are exploring is space filling if w_i are linearly independent [47]:

$$\sum_{i=1}^{d} r_i \omega_i \neq 0; -\infty < r_i < +\infty \tag{1.20}$$

Using, Ergodic theorem of Weyl [68], one can show that the statistical moments described in Equation 1.14 can be evaluated via one-dimensional integral along the curve:

$$y^p = \lim_{x \to -\infty} \frac{1}{2R} \int_{-R}^{R} f^p(x_1(s), \ldots, x_d(s))ds \tag{1.21}$$

Moreover, according to Weyl's theorem [68][47], Equation 1.14 and Equation 1.21 are equivalent:

$$\langle y^{(p)} \rangle \equiv y^{(p)} \tag{1.22}$$

Hence, for instance, model's variance:= D is given as:

$$D = \langle y^{(2)} \rangle - (\langle y^{(1)} \rangle)^2 \equiv y^{(2)} - (y^{(1)})^2 \tag{1.23}$$

And now we can compute model's variance as we have (1.21) to compute the Equation 1.23.

It is only an ideal case that the frequencies ω_i are incommensurate, that is the curve is space-filling [9]. However, such ideal case is not practically possible as the precision of computers are limited. Hence, a commensurate set of frequencies proposed by Schaibly et al. [52] can be used for the computation of the first order sensitivity indices (main effects). As the frequencies are no longer linearly independent, there is a positive R such that $f(x_1(s), \ldots, x_d(s)) = f(x_1(s + R), \ldots, x_d(s + R))$. In such case the curve becomes a closed path and the equivalency mentioned in Equation 1.22 is no longer true [47]. It has been proven by Cukier et al. [8] that $R = 2\pi$ if the frequencies ω_i are positive integers. Then we can view $f(x_1, \ldots, x_d(s))$ in this finite domain of $(-\pi, \pi)$. This redefines Equation 1.21:

$$y^{(p)} = \frac{1}{2\pi} \int_{-\pi}^{\pi} f^p(x_1(s), \ldots, x_d(s)) ds \tag{1.24}$$

Consequently the variance D of the model is redefined to be:

$$\begin{aligned} D &= y^{(2)} - (y^{(1)})^2 \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} f^2(x_1(s), \ldots, x_d(s))\, ds - \left[\frac{1}{2\pi} \int_{-\pi}^{\pi} f(x_1(s), \ldots, x_d(s))\, ds\right]^2 \end{aligned} \tag{1.25}$$

Let us for the simplicity of the notation rewrite $f(x_1(s), \ldots, x_d(s)) := f(s)$. Fourier series expansion of $f(s)$ is then:

$$y = f(s) = \sum_{j=-\infty}^{+\infty} \alpha_j \cos js + \beta_j \sin js \tag{1.26}$$

Here α_j and β_j are the Fourier coefficients and are defined as:

$$\alpha_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(s) \cos js\, ds \tag{1.27}$$

$$\beta_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(s) \sin js\, ds \tag{1.28}$$

The spectrum Θ_j of the Fourier expansion is given by:

$$\Theta_j = \alpha_j^2 + \beta_j^2 \quad \text{for} \quad j \in \mathbb{Z}, \quad \mathbb{Z} = \{-\infty, \ldots, -1, 0, 1, \ldots, +\infty\} \tag{1.29}$$

Eventually, we now know a way to approximate D_i, the proportion of the output variance D caused by the input feature X_i. To do so, the spectrum Θ_j is evaluated for the fundamental frequency ω_i and its higher harmonics $\rho\omega_i$ [9]:

$$\begin{aligned} D_i &= \sum_{\rho \in \mathbb{Z}^0} \Theta_{\rho\omega_i} \\ &= 2 \sum_{\rho=1}^{+\infty} \Theta_{\rho\omega_i} \\ &(\mathbb{Z}^0 = \mathbb{Z} - \{0\}) \end{aligned} \tag{1.30}$$

Significantly enough, Equation 1.30 is equivalent to the numerator of the Equation 1.1. To estimate the total variance of the model, we sum up all the spectrums Θ_j of the Fourier series expansion, for $j \in \mathbb{Z}^{\{0\}}$:

$$\begin{aligned} D &= \sum_{\mathbb{Z}^{\{0\}}} \Theta_j \\ &= 2\sum_{j=1}^{+\infty} \Theta_j \end{aligned} \tag{1.31}$$

Equation 1.31 is equivalent to the denominator of Equation 1.1. Moreover, by Parseval's Theorem, Equation 1.31 is also equivalent to Equation 1.25. Finally, the FAST sensitivity index $:= S_i^{FAST}$ representing the main effect of X_i, is then given as:

$$S_i^{FAST} = \frac{D_i}{D} = \frac{2\sum_{\rho=1}^{+\infty} \Theta_{\rho\omega_i}}{2\sum_{j=1}^{+\infty} \Theta_j} = \frac{\sum_{\rho=1}^{+\infty} \Theta_{\rho\omega_i}}{\sum_{j=1}^{+\infty} \Theta_j} \tag{1.32}$$

Saltelli et al. [47] extended the classical FAST of Cukier [8][9], by also incorporating the notion of total effects, as we have already mentioned in the Sobol chapter.

Consider all the frequencies that are $\notin \{\rho_1\omega_1, \ldots, \rho_d\omega_d\}$ for $\rho_i = 1, \ldots, +\infty \quad \forall i = 1, \ldots, d$. These frequencies contain the residual variance of the model $:= D_{res}$:

$$D_{\text{res}} = D - \sum_i D_i \tag{1.33}$$

D_{res} contains the variance generated by the interactions among the input features. Define the frequency of X_i as ω_i and define the frequency of all the other features as $\omega_{\neg i}$. If we then evaluate the spectrum for the frequency $\omega_{\neg i}$, and higher harmonics $\rho\omega_{\neg i}$, we can approximate the partial variance $D_{\neg i}$, representing all the effects of any order (interactions) including no contribution from X_i. Total variance caused by X_i is then [18]:

$$D_{T_i} = D - D_{\neg i} \tag{1.34}$$

Consequently, Total FAST indices, $S_{T_i}^{FAST}$ are:

$$S_{T_i}^{FAST} = \frac{D_{T_i}}{D} \tag{1.35}$$

This is also equivalent to Equation 1.12. The advantage of this algorithm is that for each X_i one only needs to opt two values for the frequencies: ω_i and $\omega_{\neg i}$. In [47], there is a detailed discussion about the way ω_i and $\omega_{\neg i}$ are selected.

To sum up about the FAST method (and its extension for the total order indices); it is a model-free (model-agnostic), variance based, global sensitivity analysis technique, just like the Sobol method. Theoretically, both Sobol and FAST are capable of measuring the high-order interactions between the input features, but some researchers assessed that FAST is not appropriate for high dimensions [39][47][69]. This is primarily due to the formulation of the fixed minimum sample size N_{min}. [8][47] is:

$$N_{min} = 2M\omega_{max} + 1 \tag{1.36}$$

Here M is the interference parameter, i.e., the number of harmonics to sum in the Fourier series decomposition (taking values usually 4-6), ω_{max} is the maximum frequence of the set ω_i. This is not the most efficient approach as the ω_{max} is an increasing function of the dimension d. Method demands more samples for higher values of d, hence potentially increasing the computational cost up to unacceptable levels [51]. Furthermore, in another study by Iooss et al. [21] it was shown that FAST performs less efficiently than Sobol for higher dimensions. To overcome such disadvantages of the FAST; RBD-FAST was proposed. We proceed with introducing the formulation of the RBD-FAST method.

RBD-FAST

The main difference in RBD-FAST method is that, unlike the complex frequency selection algorithm of the FAST method, all the input features are sampled using the same frequency ω_{fixed}, which can theoretically take any integer value up to $\frac{N-1}{2M}$. The following parametric equation ia used to obtain a sample of N points over the interval $(-\pi, \pi)$:

$$x_i = T_i(\sin \omega_{fixed} s_{ij}) \quad \forall i = 1, \ldots, d, \quad \forall j = 1, \ldots, N \tag{1.37}$$

Here $s_{i1}, s_{i2}, dots, s_{iN}$ represent the i_{th} random permutation of the N samples. Consequently Equation 1.37 generates a different random permutation for the feature x_i. We simulate the model for each sample:

$$Y(s_j) = f(x_1(s_{1j}), x_2(s_{2j}), \ldots, x_d(s_{dj})) \quad \forall j = 1, \ldots, N \tag{1.38}$$

We define $Y_{sorted}(s_j)$ in a way that: The outputted values of $Y(s_j)$ is reordered in an ascending sorted order of $x_i s_{ij}$. With such reordering, the harmonic content of x_i gets propagated to $Y_{sorted}(s_j)$ via $f(x_1(s_{1j}), x_2(s_{2j}), \ldots, x_d(s_{dj}))$ [64]. The harmonic content of such $Y_{sorted}(s_j)$ determines how much x_i affects Y, i.e., sensitivity of Y to x_i, in the following Fourier Transform:

$$F(\omega_{\text{fixed}}) = \frac{1}{\pi} \sum_{j=1}^{N} Y_{\text{sorted}}(s_j) \cdot e^{-(\text{Im}) \cdot d\omega_{\text{fixed}} \cdot s_j} \tag{1.39}$$

Note: (Im) is an imaginary number.

Finally, by evaluating Equation 1.39 at M fixed ω_{fixed} values (for simplicity: $\omega_{fixed} = 1, 2, \ldots, M$), we get D_i, the approximation of the x_i's variance contribution to the total model variance:

$$\begin{aligned} D_i &= \sum_{l=1}^{M} \left[\frac{1}{\pi} \sum_{j=1}^{N} Y_{\text{sorted}}(s_j) \cdot e^{-(\text{Im}) \cdot d\omega_{\text{fixed}} \cdot s_j} \right] \quad \Bigg| \omega_{\text{fixed}} = l \\ &= \sum_{l=1}^{M} F(\omega_{\text{fixed}}) \quad \Bigg| \omega_{\text{fixed}} = l \\ &= \sum_{l=1}^{M} F(l) \end{aligned} \tag{1.40}$$

Equation 1.40 is again equivalent to Equation 1.30 and to the denominator of Equation 1.1:

$$Var_{X_i}(E_{X_{\sim i}}(Y|X_i)) \equiv 2 \sum_{\rho=1}^{+\infty} \Theta_{\rho\omega_i} \equiv \sum_{l=1}^{M} F(l) \tag{1.41}$$

Remembering that D is the total variance of the model, the long-awaited RBD-FAST sensitivity indices, ψ_i, are defined as:

$$\psi_i = \frac{D_i}{D} \tag{1.42}$$

To summarize, the RBD-FAST method is a hybrid of RBD with classic FAST. First, the d input features are partitioned into groups of the same cardinality [64]. Then, RBD is applied independently within each group of factors. Finally, the FAST is applied between the groups, whereas a different frequency is assigned to each group. By that, RBD-FAST promises to combine RBD's computational efficiency with the FAST's accurate algorithm.

Density Based Sensitivity Analysis

In the upcoming section, we will focus on density-based methods, which examine the distribution behavior of the output rather than solely focusing on variance characterization.

Delta Moment Independent Sensitivity Measure

Motivation

So far, we have focused on variance-based methods, where the correlations of the input features were also of significant concern. For the Sobol method, in the presence of uncorrelated inputs, Oakley et al. [37] found that the model's representation (known as the Sobol decomposition) accurately represents its structure. However, the representation no longer offers the most accurate description of the model's structure when correlations between inputs become apparent. Moreover, Borgonovo [4], relying on classical utility theory, argues that the decision-maker's overall level of knowledge cannot be determined purely by variance. They further argue that since variance is merely one of the moments of the output distribution, determining which parameter reduces variance, the most does not equate to determining which parameter influences the decision-maker's state of knowledge of the output the most. A moment-independent global sensitivity method concurring with such reasoning has been proposed by Chun et al. [6]. Moment independence implies that the objective of the method is no longer the variance of the output distribution but rather the entire distribution itself. The sensitivity formulation of [6] is as following:

$$CHT_i = \frac{\sqrt{\int (y_\alpha^i - y_\alpha^0)\, d\alpha}}{E[Y^0]} \tag{1.43}$$

Here y_α^i is the α_{th} quantile if Y (model output) for the sensitivity case, and y_α^0 is the α_{th} quantile of Y for the base case. Chun-Han-Tak (CHT) measure has a concept of "sensitivity case" [6, p. 314], implying the model recomputations in certain "cases," where they are all meant to infer that our knowledge state regarding the input features has changed [[4]][6]: (a) the uncertainty range is changed; (b) the type of distribution is changed; and (c) the uncertainty associated with a feature is entirely removed. CHT_i index is an expression of Y's cumulative distribution function (CDF), F_Y. An interpretation of Equation 1.43 is that CHT_i quantifies the change in the area associated with the shift in the cumulative distribution function of Y from the base case to a sensitivity case [4].

It is good to pause and reflect on the differences between CHT_i indices [6] and the indices of the variance-based methods we have mentioned. First, CHT_i indices are based on the sensitivity case abstraction, but Sobol and RBD-FAST methods are not. Furthermore, Sobol and RBD-FAST focus on the moment (variance) of Y's distribution, but CHT_i indices do not. More specifically, CHT_i refers to the question of "which input features were (and how much) responsible for the distribution shift of Y, given a specific scenario? (e.g., uncertainty ranges of all features are reduced by a factor of 5)". Meanwhile, Sobol and RBD-FAST aim to quantify the feature contribution to $Var(Y)$ without any scenarios of hypothetical sensitivity cases.

Borgonovo [4] builds up on these concepts and proposes *Delta Moment Independent Measure* method that is moment independent, sensitivity-case independent and does not assume uncorrelated input features.

Formulation

Let $\chi = (X_1, \ldots, X_n) \in R^n$ be the set of input features. Set a function $g(\chi) : E \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}$, so that $Y = g(X)$ (g forms a functional relationship between χ and Y). Here E defines the valid range of values (measurable subset) for X in the $n - dimensional$ real space, representing a specific region or a constraint on the inputs for $g(X)$. Say $x = (x_1, \ldots, x_n)$, is the realization vector of χ. The joint cumulative distribution of the χ_i is given as $F_\chi(x)$, which also represents our state of knowledge on χ. The respective joint density of χ is given as $f_\chi(x)$. Then $f_{X_i} x_i$ is the marginal density of the realization x_i, since the marginal and joint densities are related as: $f_{X_i}(x_i) = \int \cdots \int f_\chi(x) \prod_{s \neq i} dx_s$. Regarding the output Y; $F_Y(y)$ and $f_Y(y)$ are cumulative distribution function and its corresponding density function, respectively. Finally, $f_{Y|X_i}$ expresses the conditional density of Y given one of the input features (X_i) is fixed.

With these definitions at disposal and as a next step towards definition of the moment independent importance measure, the behavior of the whole distribution of the output Y subject to constrains on input features, X_i is being observed. Figure 1.2 shows a sample example for two such densities ($f_Y(y)$, $f_{Y|X_i}$). The shift (shaded area) gets quantified by the enclosed shaded area in between:

$$s(X_i) = \int |f_Y(y) - f_{Y|X_i}(y)| dy \tag{1.44}$$

As $s(X_i)$ is a function of random variable, we can get the expected value of it as:

$$E_{X_i}[s(X_i)] = \int f_{X_i}(x_i) \left[\int |f_Y(y) - f_{Y|X_i}(y)| dy \right] dx_i \tag{1.45}$$

Eventually, the moment independent sensitivity indicator, σ_i, is defined as:

$$\sigma_i = \frac{1}{2} E_{X_i}[s(X_i)] \tag{1.46}$$

σ_i is the (moment-independent) sensitivity index of the i_{th} input feature (X_i) and represents the normalized expected shift in the distribution of Y provoked by X_i [4]. Furthermore, we can represent any group of input features as a random vector $G = (X_1, \ldots, X_t)$. Then the joint sensitivity indicator of this group:

$$\sigma_{1,\ldots,t} = \frac{1}{2} E_G[s(G)] = \int f_{X_1,\ldots,X_t}(x_1, \ldots, x_t) \times \left[\int |f_Y(y) - f_{Y|X_1,\ldots,X_t}(y)| dy \right] dx_1 \ldots dx_t \tag{1.47}$$

where

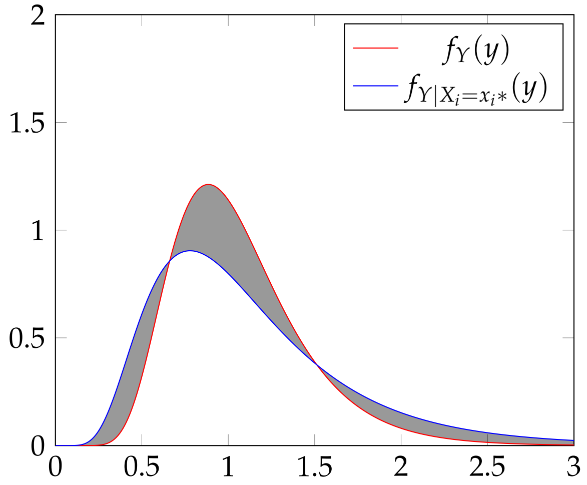


Figure 1.2: An example for shift between unconditional and conditional densities of Y. Shaded region is the shift induced by fixing input feature X_i's realization x_i*. Own Visualization.

$$\int f_{X_1,\dots,X_t}(x_1,\dots,x_t) = \int \cdots \int f_{\mathcal{X}}(x) \prod_{q \neq 1,\dots,t} dx_q \tag{1.48}$$

$f_{\mathcal{X}}(x)$ is the joint density of the set of input features as mentioned before.

Properties of σ_i:

Mathematical properties that moment-independent index σ_i possesses are arranged in Table 1.1.

Table 1.1: Properties of σ_i

	Property
1	σ_i is bounded as $0 \leq \sigma_i \leq 1$
2	If X_i does not affect Y (Y is independent of X_i), then $\sigma_i = 0$
3	Significance of all features together equals unity: $\sigma_{1,\dots,n} = 1$
4	If X_i does not affect Y (Y is independent of X_i) but X_j affects Y (Y is dependent on X_j), then: $\sigma_{ij} = \sigma_j$
5	σ_{ij} is bounded as $\sigma_i \leq \sigma_{ij} \leq \sigma_i + \sigma_{j\|i}$

PAWN

Motivation

In previous sections we have mentioned variance (Sobol Sensitivity Analysis, Random Balanced Design Fourier Amplitude Sensitivity Testing (RBD-FAST)) as well density (Delta Moment Independent Sensitivity Measure) based sensitivity analysis methods. We have already argued about the conveniences of utilizing the whole density (probability density function) information over the mere (second) moment (variance) information for the input-output relation. Pianosi et al. [41] advocate that the application of density-based methods in some fields has been limited up until now. Paper further asserts one potential explanation for the limited adoption as the higher complexity associated with implementing density-based indices compared to variance-based indices. Such complexity is primarily due to the requirement of knowing multiple conditional probability density functions (PDFs) for computation. Since PDFs are often unknown, empirical PDFs are typically used instead. The most straightforward approach involves constructing a histogram based on the data sample, but the resulting shape can be significantly influenced by the first bin's position and the bin width, making it challenging to determine appropriate values. Kernel density estimation (KDE) methods offer a more accurate approximation of PDFs by only specifying a single bandwidth parameter. Another approach involves approximating the cumulative distribution function (CDF) first and then deriving the PDF as its derivative [30]. However, the approximation procedure cannot be overly complex as computing density-based sensitivity indices often necessitates estimating numerous empirical PDFs. At a minimum, one conditional PDF per uncertain input is required, and even more are needed to consider multiple conditioning values for each input. Furthermore, the number of PDFs to be estimated becomes excessive when analyzing the accuracy or convergence of sensitivity indices, which involves computing the indices over and over using different bootstrap resamples or subsamples of varying sizes from the original dataset [41]. Expanding upon that, Piano and Wagner [41] propose the *PAWN* method for global sensitivity analysis. The main idea of *PAWN* is to describe the output distribution based on its Cumulative Distribution Function (CDF) instead of its Probability Density Function (PDF). This approach offers the benefit of approximating empirical CDFs from a data sample without any computational expenses or the need for tuning parameters. PAWN also provides an additional benefit in that sensitivity indices can be effortlessly calculated for either the entire range of output variation or a specific sub-range. This flexibility proves particularly advantageous in applications where the focus is on a certain region of the output distribution, such as the tail.

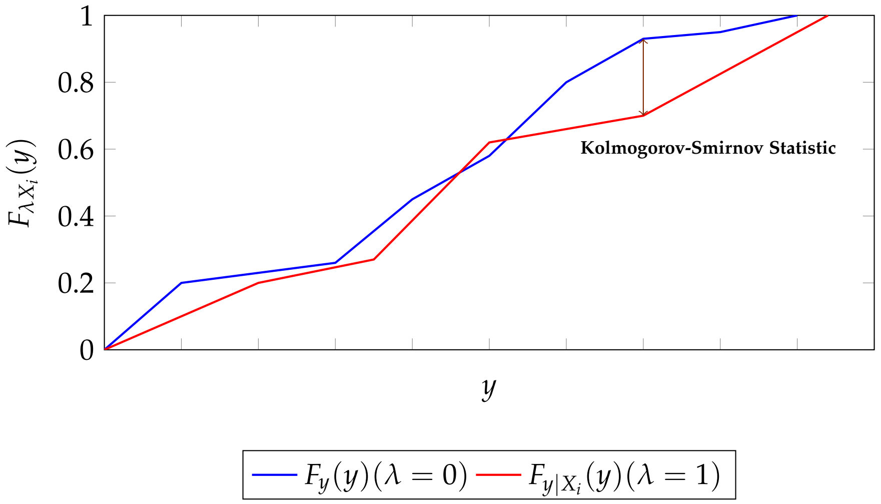


Figure 1.3: Sample Visualization of the Kolmogorov-Smirnov statistic in our context. Own Visualization.

Formulation

Suppose that the model has d features: $(X_1, \ldots, X_d)$. The proposed method comes up with a specification of PAWN index, ρ_i, which evaluates sensitivity by quantifying the changes in the output distribution that occur when the uncertainty associated with one or more input features are fixed. To be specific, the *conditional* distributions that are obtained through the variations of all input features except X_i, are compared with the *unconditional* probability distribution of the output y when all input features are allowed to vary together. $F_Y(y)$ represents the unconditional cumulative distribution of the output, whereas $F_{y|X_i}(y)$ represents the conditional cumulative distribution of the output via fixing the X_i feature. Then, Kolmogorov-Smirnov statistic [24][56] is used as a metric to quantify the difference between unconditional and conditional cumulative distribution functions:

$$Kolmogorov\_Smirnov(X_i) = \max_y \left| F_y(y) - F_{y|X_i}(y) \right| \tag{1.49}$$

A sample visualization of Kolmogorov-Smirnov statistic is given in Figure 1.3. Notice that in the figure λ plays a proxy (indicator) role for the existence of fixed X_i. That is, Kolmogorov-Smirnov statistic is dependent only on the specific fixed X_i feature. Thus PAWN index, ρ_i, is defined by a statistic (such as the maximum or the median) calculated across all possible values of X_i:

$$\rho_i = stat_{X_i}\left[Kolmogorov\_Smirnov(X_i)\right] \tag{1.50}$$

Properties of ρ_i

Properties of ρ_i are gathered in Table 1.2. First property mentions the bounds for ρ_i, which is direct consequence of the bounds of Kolmogorov-Smirnov statistic. In fact, it is one of the main reasons why this type of statistic is utilized for the PAWN analysis. Let us convince ourselves about the Property 1.

Lemma 1. *The Kolmogorov-Smirnov statistic for two CDFs is bounded between 0 and 1.*

Proof. The statistic is defined as the supremum (or maximum) of the absolute difference between the conditional distribution function, $F_{y|X_i}(y)$, and the unconditional distribution function, $F_y(y)$, evaluated at each point y:

$$D := Kolmogorov\_Smirnov(X_i) = \sup \left| F_{y|X_i}(y) - F_y(y) \right|$$

Let us first verify the lower bound,$0 \leq D$:
The conditional distribution, $F_{y|X_i}(y)$ is a function that starts at 0 and increases monotonically to 1 as y moves from negative infinity to positive infinity. On the other hand, the unconditional distribution function, $F_{y|X_i}(y)$, also ranges from 0 to 1. Therefore, at any given point y, the absolute difference $|F_{y|X_i}(y) - F_y(y)|$ can only be greater than or equal to 0. Taking the supremum (maximum) over all y, we find that the Kolmogorov-Smirnov statistic is also greater than or equal to 0. Hence, the lower bound is satisfied: $0 \leq D$.

Next, let us verify the upper bound, $D \leq 1$:
To prove the upper bound, we can consider the maximum possible difference between the two CDFs, which occurs when one CDF is 0 and the other is 1. Without loss of generality, let $F_y(y) = 0$ and $F_{y|X_i}(y) = 1$ at some point y. In this case, $|F_y(y) - F_{y|X_i}(y)| = 1$. Since D is defined as the supremum of these absolute differences over all y, it follows that $D \leq 1$.
Combining the lower and upper bounds, we conclude the proof that $0 \leq D \leq 1$. Thus, the Kolmogorov-Smirnov statistic between two different CDFs is bounded between 0 and 1.

□

Corollary. $0 \leq \rho_i \leq 1$

Proof. Since Equation 1.50, and Lemma 1, it is straightforward that any statistic such as mean, median, max, min performed over the domain of [0,1] will project to [0,1]. □

We continue with the second and third properties. As per the definition of ρ_i, the influence is quantified by the statistic over the two CDFs distance induced by the fixed X_i. Hence, if the measure over the CDFs is less, then X_i's influence is less, indeed. In an extreme case, if two CDFs coincide, then ρ_i will be 0, as a result of the Kolmogorov-Smirnov distance, which would mean that fixing X_i plays absolutely no effect on y.

Table 1.2: Properties of ρ_i

	Property
1	ρ_i is bounded as $0 \leq \sigma_i \leq 1$
2	Low ρ_i means low influence of X_i on y
3	$\rho_i = 0 \implies X_i$ is noninfluential

Numerical Implementation - Tailored Sampling Strategy [41]

The analytical solution for the Equation 1.49 (thus for obtaining ρ_i via Equation 1.50) usually does not exist. A workaround for that would be to approximate Kolmogorov-Smirnov statistic via empirical methods. An approximation would look like:

$$\widehat{Kolmogorov\_Smirnov}(X_i) = \max_y \left| \widehat{F_y(y)} - \widehat{F_{y|X_i}(y)} \right| \tag{1.51}$$

Here $\widehat{F_y(y)}$ and $\widehat{F_{y|X_i}(y)}$ are the empirical (approximations) of the unconditional and conditional CDFs, respectively. They are obtained through appropriate sampling algorithms (See 1.4 (Sampling Strategies) for a detailed discussion about sampling and more.) As soon as one decides about the setup of the sampling procedures, $\widehat{F_y(y)}$ is obtained by N number of model recomputations, each of them being originated via newly sampled input tuples. Important thing to note is that for getting $\widehat{F_y(y)}$ every feature is allowed to vary (in sampling step) within their respective spaces. For $\widehat{F_{y|X_i}(y)}$, however, M number of model recomputations performed, while only non-fixed features (every feature other than X_i) are being sampled. Following this path, we redefine (approximate) PAWN indices as:

$$\begin{aligned} \hat{\rho}_i &= stat_{X_i = \overline{X_i}^{(1)}, \overline{X_i}^{(2)}, \ldots, \overline{X_i}^{(k)}} \left[\max_y \left| \widehat{F_y(y)} - \widehat{F_{y|X_i}(y)} \right| \right] \\ &= stat_{X_i = \overline{X_i}^{(1)}, \overline{X_i}^{(2)}, \ldots, \overline{X_i}^{(k)}} \left[Kolmogorov\_Smirnov(X_i) \right] \end{aligned} \tag{1.52}$$

Here $\overline{X_i}^{(1)}, \overline{X_i}^{(2)}, \ldots, \overline{X_i}^{(k)}$ are k number of samples generated for the fixed feature X_i. The detailed process of the numerical implementation is described in Figure 1.4. Using a technique like the one in Equation 1.52 is known as a Tailored Sampling Strategy. The term "tailored" refers to the fact that a significant portion of the input samples used to calculate the sensitivity indices (ρ_i) are concentrated on specific subregions within the input variability range [42].

Enhanced Numerical Implementation - Approximation of ρ_i from any Generic Dataset [42]

In contrast to tailored sampling strategy, generic sampling techniques would distribute input samples as evenly as possible throughout the input space. Latin hypercube sampling and quasi-random sampling are just a few examples of general sampling techniques that are covered in 1.4. To approximate all PAWN sensitivity indices using the tailored sampling strategy, a model must be evaluated a total of $N + k \times M \times d$ times, d being the number of input features. The issue of how to select the triple (N, k, M) is still up for question, as it has not been formally studied and remains an open issue in the application of PAWN. However, this decision is crucial, because both the computational effort (total number of model evaluations) and accuracy of the PAWN indices are dependent on the choice of (N, k, M) [42]. Another problem with the tailored approach is that a significant portion of the computational work goes into deriving the Conditional CDFs, $\boldsymbol{F}_{Y|\sim X_i}$, which are not reusable for other uncertainty or sensitivity analysis techniques that need a generic sample. This section presents a method for approximating PAWN indices from a general dataset, addressing the problems of tailored sampling strategy.

Pianosi et al. (2018) [42] propose the method by questioning how to use Equation 1.52 with a set generated by Latin Hypercube Sampling? A solution for that is suggested as following:

- Split each X_i to n equally spaced intervals within the general bounds of x_i
- Define $\mathcal{I}_k$ as such k_{th} interval
- Redefine the conditional samples YC_i^k accordingly.
- New approximation technique for the PAWN indices is:

$$\begin{aligned} \hat{\mathcal{S}}i &= stat_{k=1,\ldots,n} \max_y |\hat{F}y(y) - \hat{F}y|X_i(y|X_i \in \mathcal{I}_k)| \\ &= stat_{k=1,\ldots,n} Kolmogorov\_Smirnov(\mathcal{I}_k) \end{aligned} \tag{1.53}$$

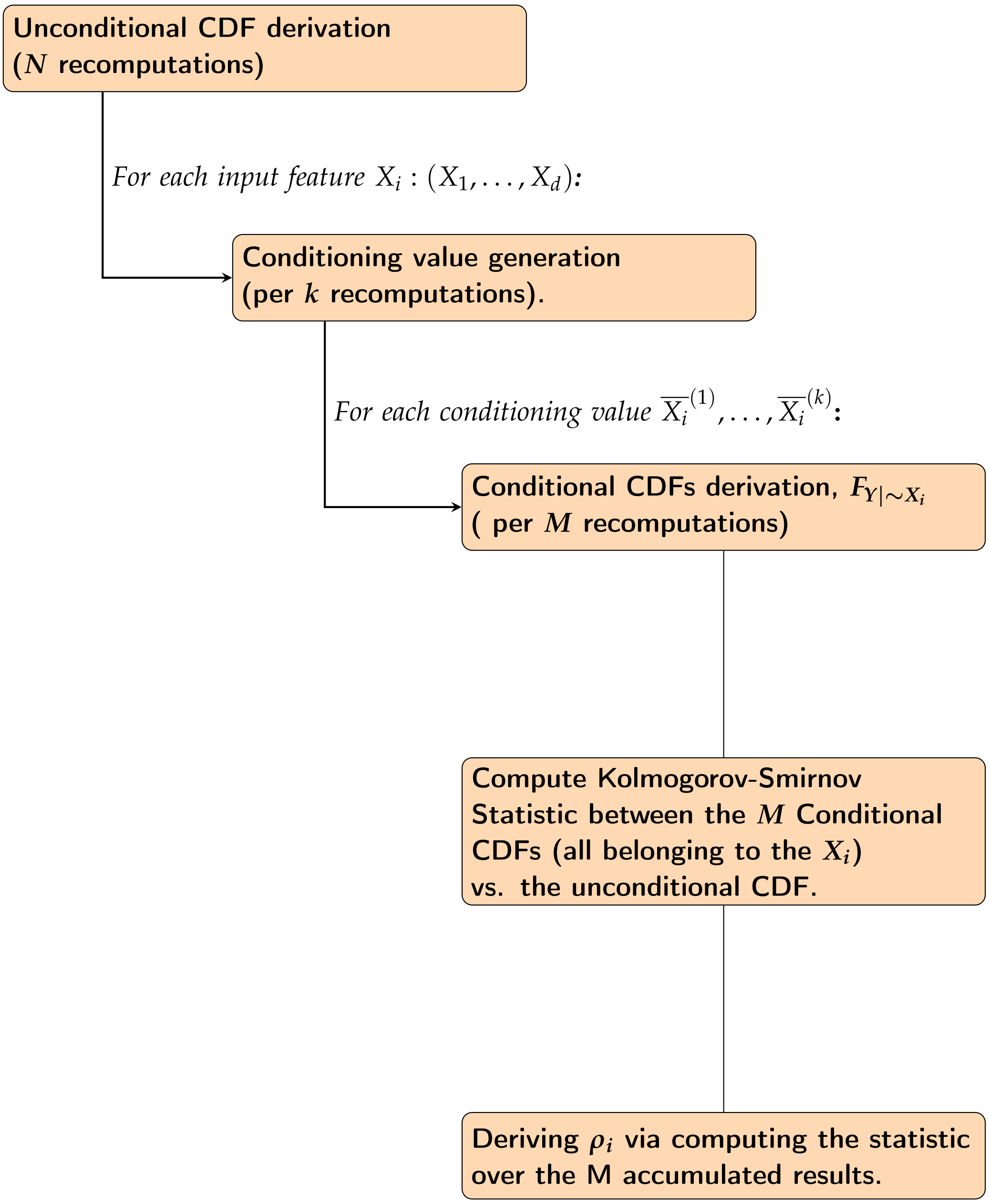

Unconditional CDF derivation
(N recomputations)
For each input feature $X_i : (X_1, \ldots, X_d)$:
Conditioning value generation
(per k recomputations).
For each conditioning value $\overline{X_i}^{(1)}, \ldots, \overline{X_i}^{(k)}$:
Conditional CDFs derivation, $F_{Y|\sim X_i}$
(per M recomputations)
Compute Kolmogorov-Smirnov
Statistic between the M Conditional
CDFs (all belonging to the X_i)
vs. the unconditional CDF.
Deriving ρ_i via computing the statistic
over the M accumulated results.

Figure 1.4: Workflow for approximations of ρ_i's using tailored sampling strategy

Using Equation 1.53 frees the user from the need of explicitly specifying M, the size of the conditional sample, because M automatically becomes the the number of points in each $\mathcal{I}_k$. For example, if the samples are uniformly spread throughout the dataset, then $M \approx \frac{N_s}{n}$, where N_s is the total number of samples and n is the number of subintervals within the bounds of X_i. In our original definition N corresponds to the number simulations for the unconditional CDF's derivation. In this approach one can either set N to N_s, or, for instance, randomly extracting a subsample with the same size as the conditional samples ($N = M$) [42].
Overall, the main contribution of this enhanced numerical implementation is that it alleviates the issue of N and M selection of the tailored sampling approach. In fact, the algorithm is fully controlled by the single value N_s, the number of samples (e.g. by Latin Hypercube Sampling) [42]. When mentioning PAWN in our methodology, we are specifically referring to this Enhanced Numerical Implementation.

1.4 Review & Mathematical Formulation of Sampling strategies

More often than not, there is no analytical solution for the calculation of sensitivity indices S_i. One remedy for that would be the approximation of the true solution by numerical methods. Numerical methods which try to reconstruct the relation between inputs and outputs rely on the concept of simulation. To be representative and comprehensive, these simulations need to encapsulate sufficient amount of input-output scenarios. Hence, adequate amount and quality of samples are needed for a decent numerical approximation performance [32][20]. In other words, sampling methodologies attempt to select the subset of individual data points, such that their common behavior estimate the characteristics of the whole population. We pay regard to a number of different sampling techniques and seek for insights and justifications for uses in our methodologies.

1.4.1 Monte Carlo Method

We start by introducing the relatively simpler method for solving the problems via sampling in probabilistic frame. The core idea is to leverage randomness to find solutions to issues that, in theory, may be deterministic. Monte Carlo methods are frequently applied to mathematical and physical issues and are particularly helpful when it is challenging or impossible to apply alternative strategies. The three problem types of optimization, numerical integration, and producing draws from a probability distribution are where Monte Carlo methods are most frequently applied [25], whereas the latter task overlaps with our study. In theory, every problem with a probabilistic interpretation can be solved using Monte Carlo methods. By applying the empirical mean of independent samples of the variable, one can approximate integrals specified by the expected value of some random variable according to the law of large numbers. Basically, the workflow in overall uncertainty quantification context (also in sensitivity analysis) is given as:

1. Get a probability frame (distribution)
2. Generate samples using a pseudo-random number generator
3. Perform deterministic computations using these samples
4. Aggregate the results and report the quantities of interest

Convergence Rate of the Monte Carlo method

Convergence rate of Monte Carlo method for N simulations is given as: $\mathcal{O}(\frac{1}{\sqrt{N}})$.

Proof. Let N be the number of simulations, μ be the true mean, σ^2 be the true variance, and $E(\cdot)$ be empirical mean. Then By the Law of Large Numbers:

$$\mu \to E(\cdot) \quad \text{as} \quad N \to \infty$$

Next, by the Central Limit Theorem, we have:

$$E(\cdot) \sim N\left(\mu, \frac{\sigma^2}{N}\right) \quad \text{as} \quad N \to \infty$$

The standard deviation of the sample mean:

$$\text{Error} = \sqrt{\frac{\sigma^2}{N}}$$

Thus, the convergence rate is given as:

$$\text{Convergence Rate} \sim \frac{1}{\sqrt{N}} \quad \text{as} \quad N \to \infty$$

□

As a positive note, the convergence rate of the algorithm does not depend on the input dimension. Moreover, the independent samples can be simulated at the same time, and that makes the algorithm "embarrasingly parallel". However, one big issue is about the significant slowness of the algorithm ($\mathcal{O}(\frac{1}{\sqrt{N}})$). As a simple scenario, if one wants to reduce the error by a factor of 10, then 100 times more simulations are needed to be performed. This roots from the pure random nature, hence, relatively poorer accuracy of the algorithm. Thus, we consider the next question: how to increase the accuracy of the Monte Carlo method? The options include:

- improving the technical details of the algorithm implementation, such as vectorizing (removing if statements), using memory efficiently etc.
- increacing N
- decreasing σ^2
- improving the sampling logic

Let us go over the listed options above. The first item clearly does not change the nature of the problem, and it has theoretical limits induced by the algorithm. The second item, increasing N, apparently (by central limit theorem) would yield more accurate approximation results. However, this approach is not desirable, e.g. high computational costs. Another approach is about decreasing the σ^2. The variance reduction methods like Control Variates [28], Importance Sampling [60], Stratified Sampling [40] are concentrated on this task. Because of its relevance to the context of sensitivity analysis[65][54][66], we will pay a special attention to the Latin Hypercube Sampling method [1], a type of Stratified Sampling. But before that, we consider the last item, improving the sampling logic, which approaches the problem via proposing a different methodology of sampling, e.g. Quasi-Monte Carlo sampling. In the next part, we are delving into the specifications of the Quasi-Monte Carlo method.

1.4.2 Quasi-Monte Carlo Method

The Quasi-Monte Carlo method incorporates low-discrepancy sequences (sub-random sequences), to solve the problems that the standard Monte Carlo method is trying to solve originally. The difference between the methods is the way the instances are sampled. We have mentioned that, the standard Monte Carlo method is based on pseudorandom number sequences. Quasi-Monte Carlo method, on the other hand, uses a "deterministic" way of going over the space. Sobol sequences [57] is an example of a low-discrepancy sequence, and will be introduced in the next part. The use of low-discrepancy sequences has the benefit of a quicker rate of convergence.

Approximating the error bounds of Quasi-Monte Carlo method

Both Monte Carlo and Quasi-Monte Carlo methods can be defined for the approximation problem of the following integration type:

$$\int_{[0,1]^s} f(u)du \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

The original integral is performed over the s-dimensional unit cube. Hence, the sample set $= \{x_1, \ldots, x_N\}$ is composed of elements such that $x_i \in \mathbb{R}^s$. The approximation error is:

$$\epsilon = |\int_{[0,1]^s} f(u)du - \frac{1}{N}\sum_{i=1}^{N} f(x_i)| \tag{1.54}$$

And according the Koksma-Hlawka inequality, it is bounded as:

$$|\epsilon| \leq V(f)D_N \tag{1.55}$$

Here V(f) defines the Hardy-Krause variation of the function f [35], whereas D_N is the discrepancy of the the set . Such discrepancy is defined as:

$$D_N = \sup_{P \subset [0,1]^s} \left| \frac{n_P}{N} - \text{volume}(P) \right| \tag{1.56}$$

Here P corresponds to a rectangular solid in $[0,1]^s$, whereas n_P is the number of its members.

Basically, given $|\epsilon| \leq V(f)D_N$, the idea of the low discrepancy sequences is minizing the error (ϵ) by decreasing D_N, which is obtained by producing the sample set whose members are "well spaced". Consequently, by Equation 1.55, it is possible to show that the error approximation (convergence rate) of the Quasi-Monte Carlo method is $\mathcal{O}(\frac{(logN)^s}{N})$ or asymptotically $\mathcal{O}(\frac{1}{N})$. Asmussen et al. have shown that a suitable low-discrepancy sequence almost always can be chosen, or the integrand can be transformed in a suitable way, to ensure that Quasi-Monte Carlo performs at least as well as Monte Carlo (and usually significantly better) [3]. Refer to the Figure 1.5, to see an example of sampling result for 500 samples in $[0,1]$. Notice the "well-spaced", and space covering character of Sobol sequences in contrast to the pseudorandom sequences. It is worthwhile to note that, Saltelli [48] has extended the Sobol sequences in order to lower error rates in the resulting sensitivity index calculations (more details can be found in [38][5][48]).

1.4.3 Latin Hypercube Sampling

As we have mentioned above, Latin Hypercube Sampling is a type of Stratified Sampling. To sample N point in a d-dimensional space, the algorithm divides each of the dimensions into N equiprobable intervals. By that, one gets N^d subcubes. The points are then sampled within each subcube so that the samples do not overlap even when projected to lower dimensions.

To be more precise, for N samples, define $\{\pi_k\}$ representing the independent random permutations of $\{1, \ldots, N\}$, for $k = 1, \ldots, n$. Each of such permutations are uniformly

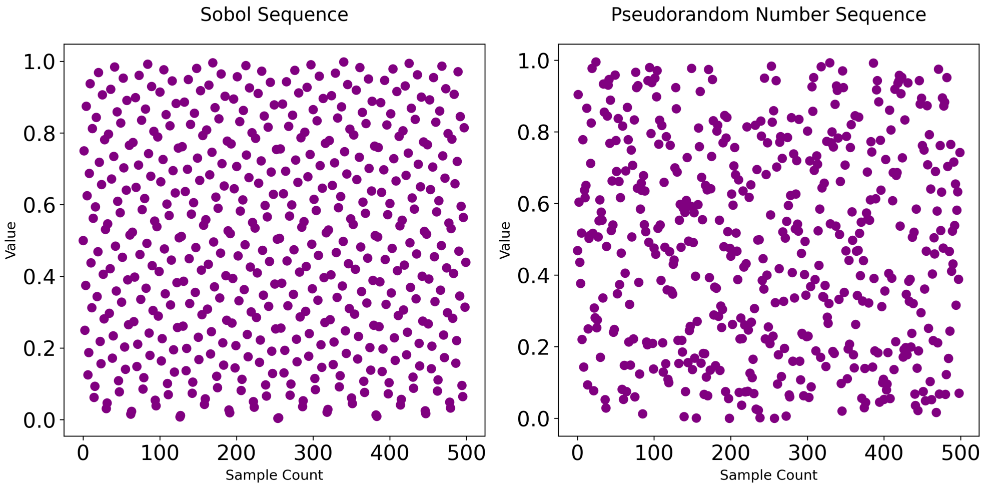


Figure 1.5: An example comparison of Sobol Sequences vs. Pseudorandom sequences for 500 samples in $[0,1]$. Own Visualization.

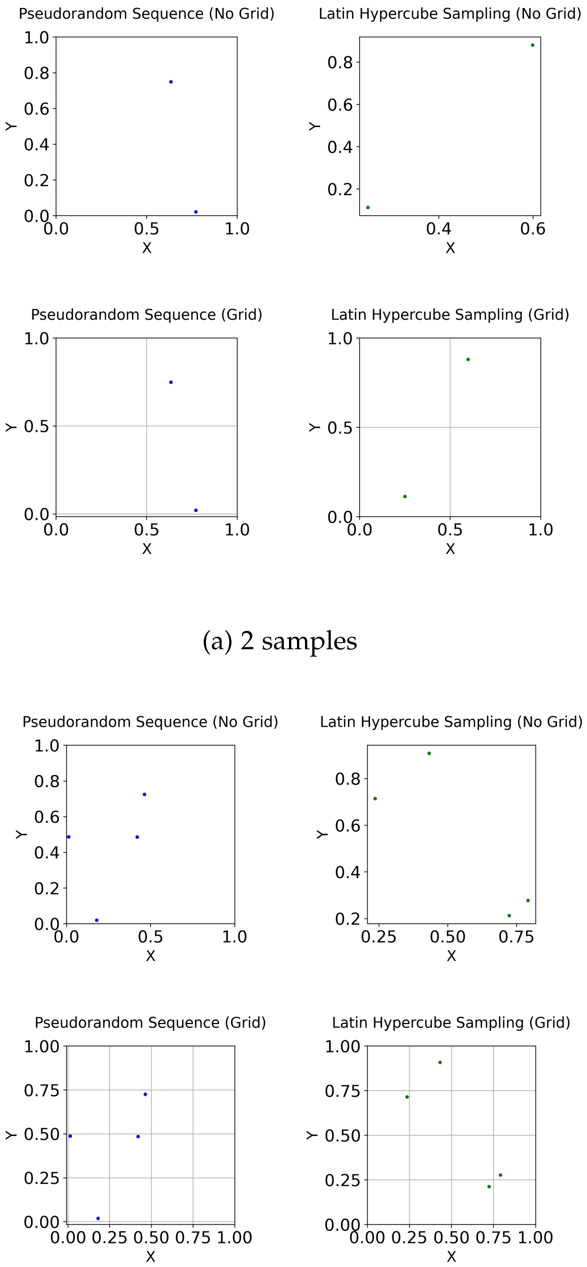
distributed over all potential $N!$ permutations [26]. The Latin Hypercube Sampling coordinates for the each sample is then given as:

$$N_i^{LHS\_coord} = \frac{\pi_k(i) - 1 + U_i^k}{N} \quad i = 1, \ldots, N \quad k = 1, \ldots, n \quad U_i^k \sim U(0,1) \tag{1.57}$$

Essentially, Latin Hypercube Sampling is designed by superimposing well stratified one-dimensional samples [26] and it tries to ensure that the coordinates for each sample are more evenly distributed to cover the input space. An alternative simpler explanation can be presented via the visualization in Figure 1.6. One can think the 2D space as a chess board and each sample as the rook. The Latin Hypercube Sampling technique ensures that, regardless of the number (say N) of samples ("rooks") placed in the sampling space ("board"), the samples are distributed in such a way that no two samples can "see" ("capture") each other when projected onto a N-grid of isoprobabilistic lines. Moreover, see the Figure 1.7 for the comparisons of the three different 1D sampling schemes for 1000 instances in $[0,1]$.

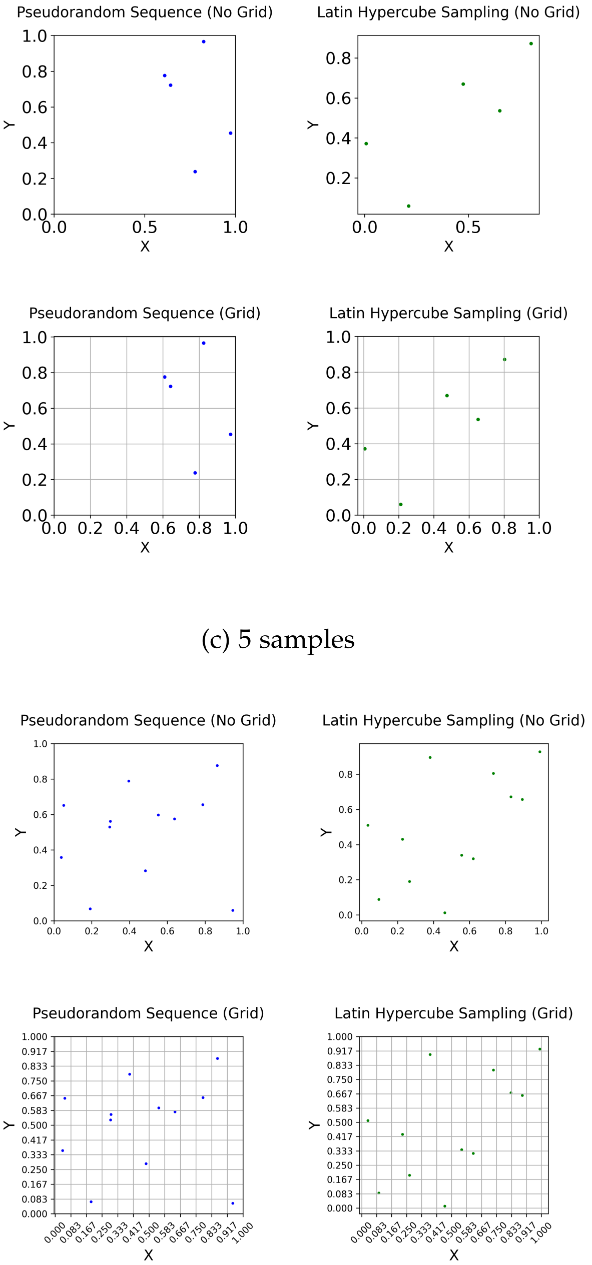
1.5 Explainable AI for Clustering step

In this section, we explore the integration of Explainable AI (XAI) techniques into the clustering process. This is an essential yet often supplementary aspect of clustering analysis, particularly when focusing on the interpretability of the results. We emphasize the use of representation learning paradigms to enhance understanding.



(a) 2 samples

(b) 4 samples



(c) 5 samples

(d) 12 samples

Figure 1.6: A "chessboard" comparison between 2D space coverage of pseudorandom sequences vs Latin Hypercube Sampling for different numbers of samples. Own Visualization.

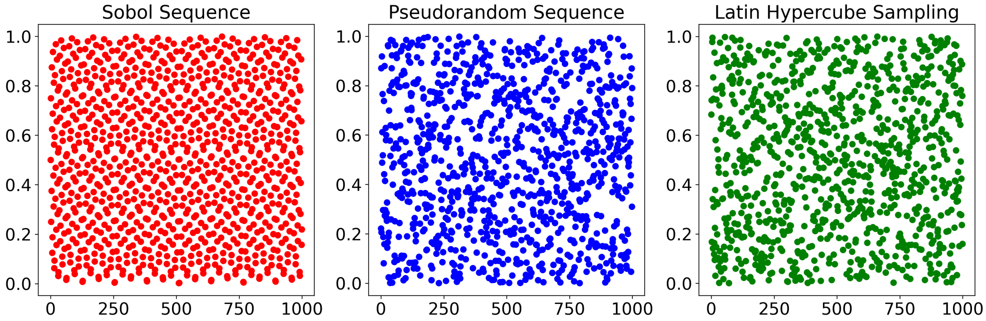


Figure 1.7: An example comparison of Sobol Sequences vs. Pseudorandom sequences vs Latin Hypercube Sampling for 1000 samples (x axis) in $[0,1]$ (y axis).Own Visualization.

For instance, consider a scenario where one seeks to understand both the overall cluster structure and the specific assignment of individual data points to clusters. This could involve analyzing embeddings generated by large language models (LLMs) or graph-based embeddings' clustering tasks. The goal here is to derive explanations that shed light on why certain data points are grouped together and how the underlying structure of the data influences these groupings. To align with the study goals of explaining the cluster results, an effective solution might be the retrieval of decision rules for the clustering process. Classification rule extraction methods offer valuable advantages when applied to the task of clustering. Although clustering is traditionally concerned with grouping similar data points into clusters, integrating such rule extraction techniques enhances the understanding and interpretability of the resulting clusters. Alternatively, a viable approach is to treat the clustering problem as a binary classification task, focusing on specific clusters and posing "this or else" (one vs. all) questions to extract cluster-specific rules. By doing so, we can obtain rules that are tailored to each cluster using the employed clustering algorithm (which can be perceived as a black-box in this context). This strategy allows for a more targeted and customized rule extraction process, enabling deeper insights into the distinct characteristics and behaviors of individual clusters.

In literature, the subject of creating (retrieving) such decision rules has received considerable attention. Several methodologies have been proposed, including Lightweight Rule Induction (LRI) by Weiss et al. [67], Maximum-Likelihood Rule Ensembles (ML-RULES) by Dembczyński et al. [11], Slipper by Cohen et al [7], RuleFit by Friedman et al. [15]. SLIPPER is a rule learner that constructs rulesets through iterative boosting of a straightforward, greedy rule-builder. It ensures that the resulting ensemble of rules remains concise and easily understandable. This is achieved by imposing constraints on the rule-builder and employing confidence-rated boosting, an extension of Adaboost [13]. Despite its simplicity, SLIPPER is both scalable and effective in learning [7]. The LRI method generates compact rules in the form of Disjunctive Normal Form (DNF). As a side note, Davey et al. [10] showed that it is theoretically possible to transform any logical formula into an equivalent DNF. However, in some cases DNF transformation can lead to exponentially more terms than the original rule set. For instance, a rule in such form: $(X_1 \vee Y_1) \wedge (X_2 \vee Y_2) \wedge \ldots \wedge (X_n \vee Y_n)$, yields 2^n terms for its DNF representation [10]. Going on with LRI method, assuming for the n-classification problem, there are an equal number of unweighted rules for each class. When classifying a new example, all rules are applied, and the class with the most satisfied rules is chosen. The induction approach seeks to decrease training error without running. Setting constraints on the size and quantity of rules specifies an overall design. During training, cases are adaptively weighted using a simple cumulative error technique. The induction approach is roughly linear in time as the number of induced rules or cases increases [67]. ML-Rules is an algorithm designed for classification tasks, specifically focused on probability estimation. In contrast to earlier rule induction methods based on sequential covering, ML-Rules takes a different approach. It treats each individual decision rule as a base classifier within an ensemble. The ensemble is constructed by iteratively minimizing the negative log-likelihood, resulting in the estimation of the class conditional probability distribution [11]. In all of the methods we have explored thus far, rule derivation relies on the process of rule induction, where each decision rule serves as the base classifier for the inductive process, allowing the creation of an ensemble of classifiers. Such ensemble is formed by iteratively minimizing a particular loss function in a greedy manner. A different, and more explanability-friendly approach is RuleFit. In RuleFit, generalized rules drawn from the data are combined linearly to create general regression and classification models. Each rule is composed of a conjunction of a few simple statements relating to the values of individual input variables. It has been demonstrated that these rule ensembles achieve predicted accuracy on par with the best techniques. But interpretation is really where they excel [15]. Each rule has an easy-to-understand structure that makes it simple to comprehend how it affects specific predictions, subgroups of predictions, or the full space of joint input variable values. Likewise, the level of significance of the relevant input variables can be evaluated globally, locally in various areas of the input space, or at specific prediction points [15]. The way those simple rules drawn from the data are combined (in a weighted way) linearly relies on solving a L1-regularized optimization problem over the weights via Gradient Directed Regularization method proposed by Friedman et al [14]. The easy-to-understand nature of this method makes it particularly favorable to use in our methodologies. Moreover, the fact that rules are being extracted from a tree ensemble, opens a door for utilizing fast algorithms such as gradient boosting and bagged decision trees. With that we also ensure the efficient implementation of the generation of such tree ensembles (weighted combination of the rules). Without loss of generality, assuming that a bagging estimator is trained for the rule ensembling, the algorithm generates multiple rules. Then, the notion of semantic deduplication is handy to ensure the reduced redundancy among the rules. In the Methods section, we describe the precise approach we employ to integrate rule extraction into the clustering process, whereas in the Results section, we showcase the obtained outcomes and implement evaluation strategies to assess the effectiveness of the generated rules.

List of Figures

List of Tables

Bibliography

[1] "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code." In: *Technometrics* 21.2 (1979), pp. 239–245. ISSN: 00401706.

[2] A. Adadi and M. Berrada. "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)." In: *IEEE Access* 6 (2018), pp. 52138–52160. DOI: `10.1109/ACCESS.2018.2870052`.

[3] S. Asmussen and P. Glynn. *Stochastic simulation: algorithms and analysis*. English. Stochastic modelling and applied probability. Netherlands: Springer, 2007. ISBN: 978-0-387-30679-7.

[4] E. Borgonovo. "A new uncertainty importance measure." In: *Reliability Engineering System Safety* 92.6 (2007), pp. 771–784. ISSN: 0951-8320. DOI: `https://doi.org/10.1016/j.ress.2006.04.015`.

[5] F. Campolongo, A. Saltelli, and J. Cariboni. "From screening to quantitative sensitivity analysis. A unified approach." In: *Computer Physics Communications* 182.4 (2011), pp. 978–988. ISSN: 0010-4655. DOI: `https://doi.org/10.1016/j.cpc.2010.12.039`.

[6] M. H. Chun, S.-J. Han, and N.-I. Tak. "An uncertainty importance measure using a distance metric for the change in a cumulative distribution function." In: *Reliab. Eng. Syst. Saf.* 70 (2000), pp. 313–321.

[7] W. W. Cohen and Y. Singer. "A Simple, Fast, and Effective Rule Learner." In: *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*. AAAI '99/IAAI '99. Orlando, Florida, USA: American Association for Artificial Intelligence, 1999, pp. 335–342. ISBN: 0262511061.

[8] R. I. Cukier, C. M. Fortuin, K. E. Shuler, A. G. Petschek, and J. H. Schaibly. "Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I Theory." In: *The Journal of Chemical Physics* 59.8 (Sept. 2003), pp. 3873–3878.

[9] R. Cukier, H. Levine, and K. Shuler. "Nonlinear sensitivity analysis of multiparameter model systems." In: *Journal of Computational Physics* 26.1 (1978), pp. 1–42. ISSN: 0021-9991. DOI: `https://doi.org/10.1016/0021-9991(78)90097-9`.

[10] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge: Cambridge University Press, 1990. ISBN: 0521365848 9780521365840 0521367662 9780521367660.

[11] K. Dembczyński, W. Kotłowski, and R. Słowiński. "Maximum Likelihood Rule Ensembles." In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, 2008, pp. 224–231. ISBN: 9781605582054. DOI: `10.1145/1390156.1390185`.

[12] F. Doshi-Velez and B. Kim. "Towards A Rigorous Science of Interpretable Machine Learning." In: *arXiv: Machine Learning* (2017).

[13] Y. Freund and R. E. Schapire. "A desicion-theoretic generalization of on-line learning and an application to boosting." In: *Computational Learning Theory*. Ed. by P. Vitányi. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 23–37. ISBN: 978-3-540-49195-8.

[14] J. Friedman and B. Popescuy. "Gradient Directed Regularization for Linear Regression and Classification." In: *Tech rep* (Jan. 2004).

[15] J. H. Friedman and B. E. Popescu. "PREDICTIVE LEARNING VIA RULE ENSEMBLES." In: *The Annals of Applied Statistics* 2 (2008), pp. 916–954.

[16] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. A. Specter, and L. Kagal. "Explaining Explanations: An Overview of Interpretability of Machine Learning." In: *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)* (2018), pp. 80–89.

[17] E. Helmers, J. Dietz, and M. Weiss. "Sensitivity Analysis in the Life-Cycle Assessment of Electric vs. Combustion Engine Cars under Approximate Real-World Conditions." In: *Sustainability* 12.3 (2020). ISSN: 2071-1050. DOI: `10.3390/su12031241`.

[18] T. Homma and A. Saltelli. "Importance measures in global sensitivity analysis of nonlinear models." In: *Reliability Engineering System Safety* 52.1 (1996), pp. 1–17. ISSN: 0951-8320. DOI: `https://doi.org/10.1016/0951-8320(96)00002-6`.

[19] E. E. Iglesias. "Sensitivity analysis of turbine engine sustainment." In: 2014.

[20] R. Iman, J. Helton, and J. Campbell. "An Approach to Sensitivity Analysis of Computer Models: Part I—Introduction, Input Variable Selection and Preliminary Variable Assessment." In: *J Qual Technol* 13 (July 1981), pp. 174–183. DOI: `10.1080/00224065.1981.11978748`.

[21] B. Iooss and P. Lemaître. "A Review on Global Sensitivity Analysis Methods." In: *Uncertainty Management in Simulation-Optimization of Complex Systems: Algorithms and Applications*. Ed. by G. Dellino and C. Meloni. Boston, MA: Springer US, 2015, pp. 101–122.

[22] M. J. Jansen. “Analysis of variance designs for model output.” In: *Computer Physics Communications* 117.1 (1999), pp. 35–43. ISSN: 0010-4655. DOI: `https://doi.org/10.1016/S0010-4655(98)00154-4`.

[23] M. Koda, G. J. Mcrae, and J. H. Seinfeld. “Automatic sensitivity analysis of kinetic mechanisms.” In: *International Journal of Chemical Kinetics* 11.4 (1979), pp. 427–444. DOI: `https://doi.org/10.1002/kin.550110408`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/kin.550110408`.

[24] A. Kolmogorov. “Sulla determinazione empirica di una legge di distribuzione.” In: *Giornale dell’Istituto Italiano degli Attuari* 4 (1933), pp. 83–91.

[25] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev. “Why the Monte Carlo method is so important today.” In: *WIREs Computational Statistics* 6.6 (2014), pp. 386–392. DOI: `https://doi.org/10.1002/wics.1314`. eprint: `https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wics.1314`.

[26] S. Kucherenko, D. Albrecht, and A. Saltelli. *Exploring multi-dimensional spaces: a Comparison of Latin Hypercube and Quasi Monte Carlo Sampling Techniques*. 2015. arXiv: `1505.02350 [stat.AP]`.

[27] Y. LeCun, Y. Bengio, and G. Hinton. “Deep Learning.” In: *Nature* 521 (May 2015), pp. 436–44. DOI: `10.1038/nature14539`.

[28] C. Lemieux. “Control Variates.” In: *Wiley StatsRef: Statistics Reference Online*. John Wiley Sons, Ltd, 2017, pp. 1–8. ISBN: 9781118445112. DOI: `https://doi.org/10.1002/9781118445112.stat07947`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118445112.stat07947`.

[29] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. “Explainable AI: A Review of Machine Learning Interpretability Methods.” In: *Entropy* 23.1 (2021). ISSN: 1099-4300. DOI: `10.3390/e23010018`.

[30] H. Liu, W. Chen, A. Sudjianto, and D. Chen. “Relative Entropy Based Method for Probabilistic Sensitivity Analysis in Engineering Design.” In: *Journal of Mechanical Design - J MECH DESIGN* 128 (May 2005). DOI: `10.1115/1.2159025`.

[31] S. Lundberg and S.-I. Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: `1705.07874 [cs.AI]`.

[32] M. D. McKay, R. J. Beckman, and W. J. Conover. “A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code.” In: *Technometrics* 21.2 (1979), pp. 239–245. ISSN: 00401706.

[33] T. Miller. “Explanation in artificial intelligence: Insights from the social sciences.” In: *Artificial Intelligence* 267 (2019), pp. 1–38. ISSN: 0004-3702. DOI: `https://doi.org/10.1016/j.artint.2018.07.007`.

[34] C. Molnar. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. 2nd ed. 2022.

[35] W. J. Morokoff and R. E. Caflisch. "Quasi-Monte Carlo Integration." In: *Journal of Computational Physics* 122.2 (1995), pp. 218–230. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.1995.1209.

[36] J. E. Oakley and A. O'Hagan. "Probabilistic Sensitivity Analysis of Complex Models: A Bayesian Approach." In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 66.3 (2004), pp. 751–769. ISSN: 13697412, 14679868.

[37] J. E. Oakley and A. O'Hagan. "Probabilistic Sensitivity Analysis of Complex Models: A Bayesian Approach." In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 66.3 (2004), pp. 751–769. ISSN: 13697412, 14679868.

[38] A. B. Owen. *On dropping the first Sobol' point*. 2021. arXiv: 2008.08051 [math.NA].

[39] Z. Pang and Z. O'Neill. "A comparison study of various sensitivity analysis methods in building applications." In: Jan. 2019. DOI: 10.26868/25222708.2019.210209.

[40] V. Parsons. "Stratified Sampling." In: Feb. 2017. ISBN: 9781118445112. DOI: 10.1002/9781118445112.stat05999.pub2.

[41] F. Pianosi and T. Wagener. "A simple and efficient method for global sensitivity analysis based on cumulative distribution functions." In: *Environmental Modelling Software* 67 (2015), pp. 1–11. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2015.01.004.

[42] F. Pianosi and T. Wagener. "Distribution-based sensitivity analysis from a generic input-output sample." In: *Environ. Model. Softw.* 108 (2018), pp. 197–207.

[43] G. Qian and A. Mahdi. "Sensitivity analysis methods in the biomedical sciences." In: *Mathematical Biosciences* 323 (2020), p. 108306. ISSN: 0025-5564. DOI: https://doi.org/10.1016/j.mbs.2020.108306.

[44] M. T. Ribeiro, S. Singh, and C. Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG].

[45] *Sensitivity Analysis of an Aircraft Engine Model Under Consideration of Dependent Variables*. Vol. Volume 1: Aircraft Engine; Fans and Blowers; Marine; Wind Energy; Scholar Lecture. Turbo Expo: Power for Land, Sea, and Air. June 2021, V001T01A005. DOI: 10.1115/GT2021-58905. eprint: https://asmedigitalcollection.asme.org/GT/proceedings-pdf/GT2021/84898/V001T01A005/6756998/v001t01a005-gt2021-58905.pdf.

[46] A. Saltelli, K. Chan, and E. Scott. *Sensitivity Analysis*. Wiley, 2009. ISBN: 9780470743829.

[47] A. Saltelli, S. Tarantola, and K. P.-S. Chan. "A Quantitative Model-Independent Method for Global Sensitivity Analysis of Model Output." In: *Technometrics* 41.1 (1999), pp. 39–56. DOI: 10.1080/00401706.1999.10485594.

[48] A. Saltelli. "Making best use of model evaluations to compute sensitivity indices." In: *Computer Physics Communications* 145.2 (2002), pp. 280–297. ISSN: 0010-4655. DOI: https://doi.org/10.1016/S0010-4655(02)00280-1.

[49] A. Saltelli and P. Annoni. "How to avoid a perfunctory sensitivity analysis." In: *Environmental Modelling Software* 25.12 (2010), pp. 1508–1517. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2010.04.012.

[50] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola. "Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index." In: *Computer Physics Communications* 181.2 (2010), pp. 259–270. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2009.09.018.

[51] F. E. Satterthwaite. "Random Balance Experimentation." In: *Technometrics* 1.2 (1959), pp. 111–137. DOI: 10.1080/00401706.1959.10489853.

[52] J. H. Schaibly and K. E. Shuler. "Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. II Applications." In: *Journal of Chemical Physics* 59 (1973), pp. 3879–3888.

[53] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization." In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. DOI: 10.1007/s11263-019-01228-7.

[54] Z. Shu and P. Jirutitijaroen. "Latin Hypercube Sampling Techniques for Power Systems Reliability Analysis With Renewable Energy Sources." In: *IEEE Transactions on Power Systems* 26.4 (2011), pp. 2066–2073. DOI: 10.1109/TPWRS.2011.2113380.

[55] K. Simonyan, A. Vedaldi, and A. Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. arXiv: 1312.6034 [cs.CV].

[56] N. Smirnov. "Estimate of deviation between empirical distribution functions in two independent samples. (Russian)." In: *Bull. Moscow University* 2(2) (1939), pp. 3–16.

[57] I. M. Sobol. "On the distribution of points in a cube and the approximate evaluation of integrals." In: *Ussr Computational Mathematics and Mathematical Physics* 7 (1967), pp. 86–112.

[58] I. M. Sobol. "Sensitivity Estimates for Nonlinear Mathematical Models." In: 1993.

[59] I. Sobol. “Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates.” In: *Mathematics and Computers in Simulation* 55.1 (2001). The Second IMACS Seminar on Monte Carlo Methods, pp. 271–280. ISSN: 0378-4754. DOI: `https://doi.org/10.1016/S0378-4754(00)00270-6`.

[60] R. Srinivasan. *Importance sampling: Applications in Communications and Detection*. Undefined. Springler Verlag, 2002. ISBN: 3-540-43420-8.

[61] B. V. Stein, E. Raponi, Z. Sadeghi, N. Bouman, R. C. H. J. Van Ham, and T. Bäck. “A Comparison of Global Sensitivity Analysis Methods for Explainable AI With an Application in Genomic Prediction.” In: *IEEE Access* 10 (2022), pp. 103364–103381. DOI: `10.1109/ACCESS.2022.3210175`.

[62] B. Sudret. “Global sensitivity analysis using polynomial chaos expansions.” In: *Reliability Engineering System Safety* 93.7 (2008). Bayesian Networks in Dependability, pp. 964–979. ISSN: 0951-8320. DOI: `https://doi.org/10.1016/j.ress.2007.04.002`.

[63] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. *Intriguing properties of neural networks*. 2014. arXiv: `1312.6199 [cs.CV]`.

[64] S. Tarantola, D. Gatelli, and T. Mara. “Random balance designs for the estimation of first order global sensitivity indices.” In: *Reliability Engineering System Safety* 91 (June 2006), pp. 717–727. DOI: `10.1016/j.ress.2005.06.003`.

[65] M. Ţene, D. E. Stuparu, D. Kurowicka, and G. Y. E. Serafy. “A copula-based sensitivity analysis method and its application to a North Sea sediment transport model.” In: *Environmental Modelling &amp Software* 104 (June 2018), pp. 1–12. DOI: `10.1016/j.envsoft.2018.03.002`.

[66] M. Vořechovský. “Extension of sample size in Latin Hypercube Sampling with correlated variables.” In: 2010.

[67] S. M. Weiss and N. Indurkhya. “Lightweight Rule Induction.” In: *Proceedings of the Seventeenth International Conference on Machine Learning*. ICML ’00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 1135–1142. ISBN: 1558607072.

[68] H. Weyl. “Mean Motion.” In: *American Journal of Mathematics* 60.4 (1938), pp. 889–896. ISSN: 00029327, 10806377.

[69] C. Xu and G. Gertner. “Understanding and comparisons of different sampling approaches for the Fourier Amplitudes Sensitivity Test (FAST).” In: *Computational Statistics Data Analysis* 55.1 (2011), pp. 184–198. ISSN: 0167-9473. DOI: `https://doi.org/10.1016/j.csda.2010.06.028`.

[70] X.-Y. Zhang, M. Trame, L. Lesko, and S. Schmidt. "Sobol Sensitivity Analysis: A Tool to Guide the Development and Evaluation of Systems Pharmacology Models." In: *CPT: Pharmacometrics Systems Pharmacology* 4 (Feb. 2015). DOI: 10.1002/psp4.6.

[71] *Computing Sensitivity Analysis of Vehicle Dynamics Based on Multibody Models*. Vol. Volume 1: 15th International Conference on Advanced Vehicle Technologies; 10th International Conference on Design Education; 7th International Conference on Micro- and Nanosystems. International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Aug. 2013, V001T01A030. DOI: 10.1115/DETC2013-13212. eprint: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/IDETC-CIE2013/55843/V001T01A030/4253986/v001t01a030-detc2013-13212.pdf.