On the Tractability of SHAP Explanations

Guy Van den Broeck,[1] **Anton Lykov,**[1] **Maximilian Schleich,**[2] **Dan Suciu**[2]
[1] University of California, Los Angeles
[2] University of Washington, Seattle

Abstract

SHAP explanations are a popular feature-attribution mechanism for explainable AI. They use game-theoretic notions to measure the influence of individual features on the prediction of a machine learning model. Despite a lot of recent interest from both academia and industry, it is not known whether SHAP explanations of common machine learning models can be computed efficiently. In this paper, we establish the complexity of computing the SHAP explanation in three important settings. First, we consider fully-factorized data distributions, and show that the complexity of computing the SHAP explanation is the same as the complexity of computing the expected value of the model. This fully-factorized setting is often used to simplify the SHAP computation, yet our results show that the computation can be intractable for commonly used models such as logistic regression. Going beyond fully-factorized distributions, we show that computing SHAP explanations is already intractable for a very simple setting: computing SHAP explanations of trivial classifiers over naive Bayes distributions. Finally, we show that even computing SHAP over the empirical distribution is #P-hard.

1 Introduction

Machine learning is increasingly applied in high stakes decision making. As a consequence, there is growing demand for the ability to explain the prediction of machine learning models. One popular explanation technique is to compute feature-attribution scores, in particular using the Shapley values from cooperative game theory (Roth 1988) as a principled aggregation measure to determine the influence of individual features on the prediction of the collective model. Shapley value based explanations have several desirable properties (Datta, Sen, and Zick 2016), which is why they have attracted a lot of interest in academia as well as industry in recent years (see e.g., Gade et al. (2019)).

Štrumbelj and Kononenko (2014) show that Shapley values can be used to explain arbitrary machine learning models. Datta, Sen, and Zick (2016) use Shapley-value-based explanations as part of a broader framework for algorithmic transparency. Lundberg and Lee (2017) use Shapley values in a framework that unifies various explanation techniques, and they coined the term SHAP explanation. They show that the SHAP explanation is effective in explaining predictions in the medical domain; see Lundberg et al. (2020). More recently there has been a lot of work on the tradeoffs of variants of the original SHAP explanations, e.g., Sundararajan and Najmi (2020), Kumar et al. (2020), Janzing, Minorics, and Bloebaum (2020), Merrick and Taly (2020), and Aas, Jullum, and Løland (2019).

Despite all of this interest, there is considerable confusion about the tractability of computing SHAP explanations. The SHAP explanations determine the influence of a given feature by systematically computing the expected value of the model given a subsets of the features. As a consequence, the complexity of computing SHAP explanations depends on the predictive model as well as assumptions on the underlying data distribution. Lundberg et al. (2020) describe a polynomial-time algorithm for computing the SHAP explanation over decision trees, but online discussions have pointed out that this algorithm is not correct as stated. We present a concrete example of this shortcoming in the supplementary material, in Appendix A. In contrast, for fully-factorized distributions, Bertossi et al. (2020) prove that there are models for which computing the SHAP explanation is #P-hard. A contemporaneous paper by Arenas et al. (2020) shows that computing the SHAP explanation for tractable logical circuits over uniform and fully factorized binary data distributions is tractable. In general, the complexity of the SHAP explanation is open.

In this paper we consider the original formulation of the SHAP explanation by Lundberg and Lee (2017) and analyze its computational complexity under the following data distributions and model classes:

1. First, we consider fully-factorized distributions, which are the simplest possible data distribution. Fully-factorized distributions capture the assumption that the model's features are independent, which is a commonly used assumption to simplify the computation of the SHAP explanations, see for example Lundberg and Lee (2017).

   For fully-factorized distributions and any prediction model, we show that the complexity of computing the SHAP explanation is the same as the complexity of computing the expected value of the model.

   It follows that there are classes of models for which the computation is tractable (e.g., linear regression, decision trees, tractable circuits) while for other models, including commonly used ones such as logistic regression and neural nets with sigmoid activation functions, it is #P-hard.

2. Going beyond fully-factorized distributions, we show that computing SHAP explanation becomes intractable already for the simplest probabilistic model that does not assume feature independence: naive Bayes. As a consequence, the complexity of computing SHAP explanations on such data distributions is also intractable for many classes of models, including linear and logistic regression.
3. Finally we consider the empirical distribution, and prove that computing SHAP explanations is #P-hard for this class of distributions. This result implies that the algorithm by Lundberg et al. (2020) cannot be fixed to compute the exact SHAP explanations over decision trees in polynomial time.

2 Background and Problem Statement

Suppose our data is described by n indexed features $\mathbf{X} = \{X_1, \dots, X_n\}$. Each feature variable X takes a value from a finite domain $\mathrm{dom}(X)$. A data instance $\mathbf{x} = (x_1, \dots, x_n)$ consists of values $x \in \mathrm{dom}(X)$ for every feature X. This instance space is denoted $\mathbf{x} \in \mathcal{X} = \mathrm{dom}(X_1) \times \cdots \times \mathrm{dom}(X_n)$. We are also given a learned function $F : \mathcal{X} \to \mathbb{R}$ that computes a prediction $F(\mathbf{x})$ on each instance $\mathbf{x}$. Throughout this paper we assume that the prediction $F(\mathbf{x})$ can be computed in polynomial time in n.

For a particular instance of prediction $F(\mathbf{x})$, the goal of *local explanations* is to clarify why the function F gave its prediction on instance $\mathbf{x}$, usually by attributing credit to the features. We will focus on local explanation that are inspired by game-theoretic Shapley values (Datta, Sen, and Zick 2016; Lundberg and Lee 2017). Specifically, we will work with the SHAP explanations as defined by Lundberg and Lee (2017).

2.1 SHAP Explanations

To produce SHAP explanations, one needs an additional ingredient: a probability distribution $\Pr(\mathbf{X})$ over the features, which we call the data distribution. We will use this distribution to reason about partial instances. Concretely, for a set of indices $S \subseteq [n] = \{1, \dots, n\}$, we let $\mathbf{x}_S$ denote the restriction of complete instance $\mathbf{x}$ to those features $\mathbf{X}_S$ with indices in S. Abusing notation, we will also use $\mathbf{x}_S$ to denote the probabilistic event $\mathbf{X}_S = \mathbf{x}_S$.

Under this data distribution, it now becomes possible to ask for the *expected value* of the predictive function F. Clearly, for a complete data instance $\mathbf{x}$ we have that $\mathbf{E}[F|\mathbf{x}] = F(\mathbf{x})$, as there is no uncertainty about the features. However, for a partial instance $\mathbf{x}_S$, which does not assign values to the features outside of $\mathbf{X}_S$, we appeal to the data distribution Pr to compute the expectation of function F as $\mathbf{E}_{\Pr}[F|\mathbf{x}_S] = \sum_{\mathbf{x} \in \mathcal{X}} F(\mathbf{x}) \Pr(\mathbf{x}|\mathbf{x}_S)$.

The SHAP explanation framework draws from Shapley values in cooperative game theory. Given a particular instance $\mathbf{x}$, it considers features $\mathbf{X}$ to be players in a coalition game: the game of making a prediction for $\mathbf{x}$. SHAP explanations are defined in terms of a set function $v_{F,\mathbf{x},\Pr} : 2^{\mathbf{X}} \to \mathbb{R}$. Its purpose is to evaluate the *"value"* of each coalition of players/features $\mathbf{X}_S \subseteq \mathbf{X}$ in making the prediction $F(\mathbf{x})$ under data distribution Pr. Concretely, following Lundberg and Lee (2017), this value function is the conditional expectation of function F:

$$\mathrm{v}_{F,\mathbf{x},\Pr}(\mathbf{X}_S) \overset{\text{def}}{=} \mathbf{E}_{\Pr}[F|\mathbf{x}_S]. \tag{1}$$

We will elide F, $\mathbf{x}$, and Pr when they are clear from context.

Our goal, however, is to assign credit to individual features. In the context of a coalition $\mathbf{X}_S$, the *contribution* of an individual feature $X \notin \mathbf{X}_S$ is given by

$$\mathrm{c}(X, \mathbf{X}_S) \overset{\text{def}}{=} \mathrm{v}(\mathbf{X}_S \cup \{X\}) - \mathrm{v}(\mathbf{X}_S). \tag{2}$$

where each term is implicitly w.r.t. the same F, $\mathbf{x}$, and Pr.

Finally, the SHAP explanation computes a score for each feature $X \in \mathbf{X}$ averaged over all possible contexts, and thus measures the influence feature X has on the outcome. Let π be a permutation on the set of features $\mathbf{X}$, i.e., π fixes a total order on all features. Let $\pi^{<X}$ be the set of features that come before X in the order π. The SHAP explanations are then defined as computing the following scores.

Definition 1 (SHAP Score)**.** Fix an entity $\mathbf{x}$, a predictive function F, and a data distribution Pr. The SHAP explanation of a feature X is the contribution of X given the features $\pi^{<X}$, averaged over all permutations π:

$$\textsc{Shap}(X) \overset{\text{def}}{=} \frac{1}{n!} \sum_{\pi} \mathrm{c}(X, \pi^{<X}). \tag{3}$$

We mention two simple properties of the SHAP explanations here; for more discussion see Datta, Sen, and Zick (2016) and Lundberg et al. (2020). First, for the linear combination of functions $G(.) = \sum_k \lambda_k F_k(.)$, we have that

$$\textsc{Shap}_G(X) = \sum_k \lambda_k \textsc{Shap}_{F_k}(X). \tag{4}$$

Second, the sum of the SHAP explanation of all features is related to the expected value of function F:

$$\sum_i \textsc{Shap}_F(X_i) = F(\mathbf{x}) - \mathbf{E}[F]. \tag{5}$$

2.2 Computational Problems

This paper studies the complexity of computing $\textsc{Shap}(X)$; a task we formally define next. We write $\mathtt{F}$ for a class of functions. We also write $\mathtt{PR}_n$ for a class of data distributions over n features, and let $\mathtt{PR} = \bigcup_n \mathtt{PR}_n$. We assume that all parameters are rationals. Because SHAP explanations are for an arbitrary fixed instance $\mathbf{x}$, we will simplify the notation throughout this paper by assuming it to be the instance $\mathbf{e} = (1, 1, \dots, 1)$, and that each domain contains the value 1, which is without loss of generality.

Definition 2 (SHAP Computational Problems)**.** For each function class $\mathtt{F}$ and distribution class $\mathtt{PR}$, consider the following computational problems.

- The *functional* SHAP *problem* $\mathtt{F\text{-}SHAP(F,PR)}$: given a data distribution $\Pr \in \mathtt{PR}$ and a function $F \in \mathtt{F}$, compute $\textsc{Shap}(X_1), \dots, \textsc{Shap}(X_n)$.

– The *decision* SHAP *problem* D-SHAP(F, PR): given a data distribution $\Pr \in$ PR, a function $F \in$ F, a feature $X \in \mathbf{X}$, and a threshold $t \in \mathbb{R}$, decide if SHAP$(X) > t$.

To establish the complexities of these problems, we use standard notions of reductions. A *polynomial time reduction* from a problem A to a problem B, denoted by A $\leq^P$ B, and also called a *Cook reduction*, is a polynomial-time algorithm for the problem A with access to an oracle for the problem B. We write A $\equiv^P$ B when both A $\leq^P$ B and B $\leq^P$ A.

In the remainder of this paper will study the computational complexity of these problems for natural hypothesis classes F that are popular in machine learning, as well as common classes of data distributions PR, including those most often used to compute SHAP explanations.

3 SHAP over Fully-Factorized Distributions

We start our study of the complexity of SHAP by considering the simplest probability distribution: a fully-factorized distribution, where all features are independent.

There are both practical and computational reasons why it makes sense to assume a fully-factorized data distribution when computing SHAP explanations. First, functions F are often the product of a supervised learning algorithm that does not have access to a generative model of the data – it is purely discriminative. Hence, it is convenient to make the practical assumption that the data distribution is fully factorized, and therefore easy to estimate. Second, fully-factorized distributions are highly tractable; for example they make it easy to compute expectations of linear regression functions (Khosravi et al. 2019b) and other hard inference tasks (Vergari et al. 2020).

Lundberg and Lee (2017) indeed observe that computing the SHAP-explanation on an arbitrary data distribution is challenging and consider using fully-factorized distributions (Sec. 4, Eq. 11). Other prior work on computing explanations also use fully-factorized distributions of features, e.g., Datta, Sen, and Zick (2016); Štrumbelj and Kononenko (2014). As we will show, the SHAP explanation can be computed efficiently for several popular classifiers when the distribution is fully factorized. Yet, such simple data distributions are not a guarantee for tractability: computing SHAP scores will be intractable for some other common classifiers.

3.1 Equivalence to Computing Expectations

Before studying various function classes, we prove a key result that connects the complexity of SHAP explanations to the complexity of computing expectations.

Let IND$_n$ be the class of fully-factorized probability distributions over n discrete and independent random variables $X_1, \ldots, X_n$. That is, for every instance $(x_1, \ldots, x_n) \in \mathcal{X}$, we have that $\Pr(X_1 = x_1, \ldots, X_n = x_n) = \prod_i \Pr(X_i = x_i)$. Let IND $\stackrel{\text{def}}{=} \bigcup_{n \geq 0}$ IND$_n$. We show that for every function class F, the complexity of F-SHAP(F, IND) is the same as that of the *fully-factorized expectation problem.*

Definition 3 (Fully-Factorized Expectation Problem)**.** Let F be a class of real-valued functions with discrete inputs. The *fully-factorized expectation problem* for F, denoted E(F), is the following: given a function $F \in$ F and a probability distribution $\Pr \in$ IND, compute $\mathbf{E}_{\Pr}(F)$.

We know from Equation 5 that for any function F over n features, E$(\{F\}) \leq^P$ F-SHAP$(\{F\}, \text{IND}_n)$, because $\mathbf{E}[F] = F(\mathbf{x}) - \sum_{i=1,n} \text{SHAP}_F(X_i)$. In this section we prove that the converse holds too:

Theorem 1. *For any function* $F : \mathcal{X} \to \mathbb{R}$*, we have that* F-SHAP$(\{F\}, \text{IND}_n) \equiv^P$ E$(\{F\})$.

In other words, for *any* function F, the complexity of computing the SHAP scores is the same as the complexity of computing the expected value $\mathbf{E}[F]$ under a fully-factorized data distribution. One direction of the proof is immediate: E$(\{F\}) \leq^P$ F-SHAP$(\{F\}, \text{IND}_n)$ because, if we are given an oracle to compute SHAP$_F(X_i)$ for every feature X_i, then we can obtain $\mathbf{E}[F]$ from Equation 5 (recall that we assumed that $F(\mathbf{x})$ is computable in polynomial time). The hard part of the proof is the opposite direction: we will show in Sec. 3.2 how to compute SHAP$_F(X_i)$ given an oracle for computing $\mathbf{E}[F]$. Theorem 1 immediately extends to classes of functions F, and to any number of variables, and therefore implies that F-SHAP(F, IND) $\equiv^P$ E(F).

Sections 3.3 and 3.4 will discuss the consequences of this result, by delineating which function classes support tractable SHAP explanations, and which do not. The next section is devoted to proving our main technical result.

3.2 Proof of Theorem 1

We start with the special case when all features X are binary: $\text{dom}(X) = \{0, 1\}$. We denote by INDB$_n$ the class of fully-factorized distributions over binary domains.

Theorem 2. *For any function* $F : \{0,1\}^n \to \mathbb{R}$*, we have that* F-SHAP$(\{F\}, \text{INDB}_n) \equiv^P$ E$(\{F\})$.

Proof. We prove only F-SHAP$(F, \text{INDB}_n) \leq^P$ E$(\{F\})$; the opposite direction follows immediately from Equation 5. We will assume w.l.o.g. that F has $n + 1$ binary features $\mathbf{X}' = \{X_0\} \cup \mathbf{X}$ and show how to compute SHAP$_F(X_0)$ using repeated calls to an oracle for computing $\mathbf{E}[F]$, i.e., the expectation of the same function F, but over fully-factorized distributions with different probabilities. The probability distribution Pr is given to us by $n + 1$ rational numbers, $p_i \stackrel{\text{def}}{=} \Pr(X_i = 1)$, $i = 0, n$; obviously, $\Pr(X_i = 0) = 1 - p_i$. Recall that the instance whose outcome we want to explain is $\mathbf{e} = (1, \ldots, 1)$. Recall that for any set $S \subseteq [n]$ we write $\mathbf{e}_S$ for the event $\bigwedge_{i \in S}(X_i = 1)$. Then, we have that

$$\text{SHAP}(X_0) = \sum_{k=0,n} \frac{k!(n-k)!}{(n+1)!} D_k, \quad \text{where} \tag{6}$$

$$D_k \stackrel{\text{def}}{=} \sum_{S \subseteq [n] : |S| = k} \left(\mathbf{E}\left[F|\mathbf{e}_{S \cup \{0\}}\right] - \mathbf{E}[F|\mathbf{e}_S]\right).$$

Let $F_0 \stackrel{\text{def}}{=} F[X_0 := 0]$ and $F_1 \stackrel{\text{def}}{=} F[X_0 := 1]$ (both are functions in n binary features, $\mathbf{X} = \{X_1, \ldots, X_n\}$). Then:

$$\mathbf{E}\left[F|\mathbf{e}_{S \cup \{0\}}\right] = \mathbf{E}[F_1|\mathbf{e}_S]$$
$$\mathbf{E}[F|\mathbf{e}_S] = \mathbf{E}[F_0|\mathbf{e}_S] \cdot (1 - p_0) + \mathbf{E}[F_1|\mathbf{e}_S] \cdot p_0$$

and therefore D_k is given by:

$$D_k = (1 - p_0) \sum_{S\subseteq[n]:|S|=k} (\mathbf{E}[F_1|\mathbf{e}_S] - \mathbf{E}[F_0|\mathbf{e}_S])$$

For any function G, Equation 1 defines value $\mathrm{v}_{G,\mathbf{e},\Pr}(\mathbf{X}_S)$ as $\mathbf{E}[G|\mathbf{e}_S]$. Abusing notation, we write $v_{G,k}$ for the sum of these quantities over all sets S of cardinality k:

$$v_{G,k} \overset{\text{def}}{=} \sum_{S\subseteq[n],|S|=k} \mathbf{E}[G|\mathbf{e}_S]. \tag{7}$$

We will prove the following claim.

Claim 1. Let G be a function over n binary variables. Then the $n+1$ quantities $v_{G,0}$ until $v_{G,n}$ can be computed in polynomial time, using $n+1$ calls to an oracle for $\mathtt{E}(\{G\})$.

Note that an oracle for $\mathtt{E}(\{F\})$ is also an oracle for both $\mathtt{E}(\{F_0\})$ and $\mathtt{E}(\{F_1\})$, by simply setting $\Pr(X_0 = 1) = 0$ or $\Pr(X_0 = 1) = 1$ respectively. Therefore, Claim 1 proves Theorem 2, by applying it once to F_0 and once to F_1 in order to derive all the quantities $v_{F_0,k}$ and $v_{F_1,k}$, thereby computing D_k, and finally computing $\textsc{Shap}_F(X_0)$ using Equation 6. It remains to prove Claim 1.

Fix a function G over n binary variables and let $v_k = v_{G,k}$. Let $p_j = \Pr(X_j = 1)$, for $j = 1, n$, define the distribution over which we need to compute $v_0, \ldots, v_n$. We will prove the following additional claim.

Claim 2. Given any real number $z > 0$, consider the distribution $\Pr_z(X_j) = p'_j \overset{\text{def}}{=} \frac{p_j+z}{1+z}$, for $j = 1, n$. Let $\mathbf{E}_z[G]$ denote $\mathbf{E}[G]$ under distribution $\Pr_z$. We then have that

$$\sum_{k=0,n} z^k \cdot v_k = (1+z)^n \cdot \mathbf{E}_z[G]. \tag{8}$$

Assuming Claim 2 holds, we prove Claim 1. Choose any $n+1$ distinct values for z, use the oracle to compute the quantities $\mathbf{E}_{z_0}[G], \ldots, \mathbf{E}_{z_n}[G]$, and form a system of $n+1$ linear equations (8) with unknowns $v_0, \ldots, v_n$. Next, observe that its matrix is a non-singular Vandermonde matrix, hence the system has a unique solution which can be computed in polynomial time. It remains to prove Claim 2.

Because of independence, the probability of instance $\mathbf{x} \in \{0,1\}^n$ is $\Pr(\mathbf{x}) = \prod_{i:\mathbf{x}_i=1} p_i \cdot \prod_{i:\mathbf{x}_i=0}(1 - p_i)$, where $\mathbf{x}_i$ looks up the value of feature X_i in instance $\mathbf{x}$. Similarly, $\Pr_z(\mathbf{x}) = \prod_{i:\mathbf{x}_i=1} p'_i \cdot \prod_{i:\mathbf{x}_i=0}(1 - p'_i)$. Using direct calculations we derive:

$$\Pr(\mathbf{x}) \prod_{i:\mathbf{x}_i=1} \left(1 + \frac{z}{p_i}\right) = (1+z)^n \cdot \Pr_z(\mathbf{x}) \tag{9}$$

Separately we also derive the following identity, using the fact that $\Pr(\mathbf{e}_S) = \prod_{i\in S} p_i$ by independence:

$$\mathbf{E}[G|\mathbf{e}_S] = \frac{1}{\prod_{i\in S} p_i} \sum_{\mathbf{x}:\mathbf{x}_S=\mathbf{e}_S} G(\mathbf{x}) \cdot \Pr(\mathbf{x}) \tag{10}$$

We are now in a position to prove Claim 2:

$$\begin{aligned}
\sum_{k=0,n} z^k \cdot v_k &= \sum_{k=0,n} z^k \sum_{S\subseteq[n]:|S|=k} \mathbf{E}[G|\mathbf{e}_S] \\
&= \sum_{S\subseteq[n]} z^{|S|} \cdot \mathbf{E}[G|\mathbf{e}_S] \\
&= \sum_{S\subseteq[n]} \frac{z^{|S|}}{\prod_{i\in S} p_i} \sum_{\mathbf{x}:\mathbf{x}_S=\mathbf{e}_S} G(\mathbf{x}) \cdot \Pr(\mathbf{x})
\end{aligned}$$

The last line follows from Equation 10. Next, we simply exchange the summations $\sum_S$ and $\sum_{\mathbf{x}}$, after which we apply the identity $\sum_{S\subseteq A} \prod_{i\in S} u_i = \prod_{i\in A}(1 + u_i)$.

(continuing)

$$\begin{aligned}
&= \sum_{\mathbf{x}\in\{0,1\}^n} G(\mathbf{x}) \cdot \Pr(\mathbf{x}) \sum_{S:\mathbf{x}_S=\mathbf{e}_S} \frac{z^{|S|}}{\prod_{i\in S} p_i} \\
&= \sum_{\mathbf{x}\in\{0,1\}^n} G(\mathbf{x}) \cdot \Pr(\mathbf{x}) \prod_{i:\mathbf{x}_i=1} \left(1 + \frac{z}{p_i}\right) \\
&= (1+z)^n \sum_{\mathbf{x}\in\{0,1\}^n} G(\mathbf{x}) \cdot \Pr_z(\mathbf{x}) = (1+z)^n \cdot \mathbf{E}_z[G].
\end{aligned}$$

The final line uses Equation 9. This completes the proof of Claim 2 as well as Theorem 2. □

Next, we generalize this result from binary features to arbitrary discrete features. Fix a function with n inputs, $F : \mathcal{X}(\overset{\text{def}}{=} \prod_i \mathrm{dom}(X_i)) \to \mathbb{R}$, where each domain is an arbitrary finite set, $\mathrm{dom}(X_i) = \{1, 2, \ldots, m_i\}$; we assume w.l.o.g. that $m_i > 1$. A fully factorized probability space $\Pr \in \mathtt{IND}_n$ is defined by numbers $p_{ij} \in [0,1]$, $i = 1, n$, $j = 1, m_i$, such that, for all i, $\sum_j p_{ij} = 1$. Given F and $\Pr$ over the domain $\prod_i \mathrm{dom}(X_i)$, we define their *projections*, $F_\pi, \Pr_\pi$ over the binary domain $\{0,1\}^n$ as follows. For any instance $\mathbf{x} \in \{0,1\}^n$, let $T(\mathbf{x})$ denote the event asserting that $X_j = 1$ iff $\mathbf{x}_j = 1$. Formally,

$$T(\mathbf{x}) \overset{\text{def}}{=} \bigwedge_{j:\mathbf{x}_j=1} (X_j = 1) \wedge \bigwedge_{j:\mathbf{x}_j=0} (X_j \neq 1).$$

Then, the projections are defined as follows: $\forall \mathbf{x} \in \{0,1\}^n$,

$$\Pr_\pi(\mathbf{x}) \overset{\text{def}}{=} \Pr(T(\mathbf{x})) \qquad F_\pi(\mathbf{x}) \overset{\text{def}}{=} \mathbf{E}[F \mid T(\mathbf{x})] \tag{11}$$

Notice that F_π depends both on F and on the probability distribution $\Pr$. Intuitively, the projections only distinguishes between $X_j = 1$ and $X_j \neq 1$, for example:

$$\begin{aligned}
F_\pi(1,0,0) =& \mathbf{E}[F|(X_1 = 1, X_2 \neq 1, X_3 \neq 1)] \\
\Pr_\pi(1,0,0) =& \Pr(X_1 = 1, X_2 \neq 1, X_3 \neq 1)
\end{aligned}$$

We prove the following result in Appendix B:

Proposition 3. *Let $F : \mathcal{X} \to \mathbb{R}$ be a function with n input features, and $\Pr \in \mathtt{IND}_n$ a fully factorized distribution over $\mathcal{X}$. Then (1) for any feature X_j, $\textsc{Shap}_{F,\Pr}(X_j) = \textsc{Shap}_{F_\pi,\Pr_\pi}(X_j)$, and (2) $\mathtt{E}(\{F_\pi\}) \leq^P \mathtt{E}(\{F\})$.*

Item (1) states that the SHAP-score of F computed over the probability space Pr is the same as that of its projection F_π (which depends on Pr) over the projected probability space $\Pr_\pi$. Item (2) says that, for any probability space over $\{0,1\}^n$ (not necessarily $\Pr_\pi$), we can compute $\mathbf{E}[F_\pi]$ in polynomial time given access to an oracle for computing $\mathbf{E}[F]$. We can now complete the proof of Theorem 1, by showing that $\texttt{F-SHAP}(\{F\}, \texttt{IND}_n) \leq^P \texttt{E}(\{F\})$. Given a function F and probability space $\Pr \in \texttt{IND}_n$, in order to compute $\text{SHAP}_{F,\Pr}(X_j)$, by item (1) of Proposition 3 it suffices to show how to compute $\text{SHAP}_{F_\pi,\Pr_\pi}(X_j)$. By Theorem 2, we can compute the latter given access to an oracle for computing $\mathbf{E}[F_\pi]$. Finally, by item (2) of the proposition, we can compute $\mathbf{E}[F_\pi]$ given an oracle for computing $\mathbf{E}[F]$.

3.3 Tractable Function Classes

Given the polynomial-time equivalence between computing SHAP explanations and computing expectations under fully-factorized distributions, a natural next question is: which real-world hypothesis classes in machine learning support efficient computation of SHAP scores?

Corollary 4. *For the following function classes* $\texttt{F}$*, computing* SHAP *scores* $\texttt{F-SHAP}(\texttt{F}, \texttt{IND})$ *is in polynomial time in the size of the representations of function* $F \in \texttt{F}$ *and fully-factorized distribution* $\Pr \in \texttt{IND}$.

1. *Linear regression models*
2. *Decision and regression trees*
3. *Random forests or additive tree ensembles*
4. *Factorization machines, regression circuits*
5. *Boolean functions in d-DNNF, binary decision diagrams*
6. *Bounded-treewidth Boolean functions in CNF*

These are all consequences of Theorem 1, and the fact that computing fully-factorized expectations $\texttt{E}(\texttt{F})$ for these function classes $\texttt{F}$ is in polynomial time. Concretely, we have the following observations about fully-factorized expectations:

1. Expectations of linear regression functions are efficiently computed by mean imputation (Khosravi et al. 2019b). The tractability of SHAP on linear regression models is well known. In fact, Štrumbelj and Kononenko (2014) provide a closed-form formula for this case.
2. Paths from root to leaf in a decision or regression tree are mutually exclusive. Their expected value is therefore the sum of expected values of each path, which are tractable to compute within $\texttt{IND}$; see Khosravi et al. (2020).
3. Additive mixtures of trees, as obtained through bagging or boosting, are tractable, by the linearity of expectation.
4. Factorization machines extend linear regression models with feature interaction terms and factorize the parameters of the higher-order terms (Rendle 2010). Their expectations remain easy to compute. Regression circuits are a graph-based generalization of linear regression. Khosravi et al. (2019a) provide an algorithm to efficiently take their expectation w.r.t. a probabilistic circuit distribution, which is trivial to construct for the fully-factorized case.

The remaining tractable cases are Boolean functions. Computing fully-factorized expectations of Boolean functions is widely known as the *weighted model counting* task (WMC) (Sang, Beame, and Kautz 2005; Chavira and Darwiche 2008). WMC has been extensively studied both in the theory and the AI communities, and the precise complexity of $\texttt{E}(\texttt{F})$ is known for many families of Boolean functions $\texttt{F}$. These results immediately carry over to the $\texttt{F-SHAP}(\texttt{F}, \texttt{IND})$ problem through Theorem 1:

5. Expectations can be computed in time linear in the size of various circuit representations, called d-DNNF, which includes binary decision diagrams (OBDD, FBDD) and SDDs (Bryant 1986; Darwiche and Marquis 2002).[1]
6. Bounded-treewidth CNFs are efficiently compiled into OBDD circuits (Ferrara, Pan, and Vardi 2005), and thus enjoy tractable expectations.

To conclude this section, the reader may wonder about the algorithmic complexity of solving $\texttt{F-SHAP}(\texttt{F}, \texttt{IND})$ with an oracle for $\texttt{E}(\texttt{F})$ under the reduction in Section 3.2. Briefly, we require a linear number of calls to the oracle, as well as time in $O(n^3)$ for solving a system of linear equations. Hence, for those classes, such as d-DNNF circuits, where expectations are linear in the size of the (circuit) representation of F, computing $\texttt{F-SHAP}(\texttt{F}, \texttt{IND})$ is also linear in the representation size and polynomial in n.

3.4 Intractable Function Classes

The polynomial-time equivalence of Theorem 1 also implies that computing SHAP scores must be intractable whenever computing fully-factorized expectations is intractable. This section reviews some of those function classes $\texttt{F}$, including some for which the computational hardness of $\texttt{E}(\texttt{F})$ is well known. We begin, however, with a more surprising result.

Logistic regression is one of the simplest and most widely used machine learning models, yet it is conspicuously absent from Corollary 4. We prove that computing the expectation of a logistic regression model is #P-hard, even under a uniform data distribution, which is of independent interest.

A *logistic regression* model is a parameterized function $F(\boldsymbol{x}) \overset{\text{def}}{=} \sigma(\boldsymbol{w} \cdot \boldsymbol{x})$, where $\boldsymbol{w} = (w_0, w_1, \ldots, w_n)$ is a vector of weights, $\sigma(z) = 1/(1+e^{-z})$ is the logistic function, $\boldsymbol{x} \overset{\text{def}}{=} (1, x_1, x_2, \ldots, x_n)$, and $\boldsymbol{w} \cdot \boldsymbol{x} \overset{\text{def}}{=} \sum_{i=0,n} w_i x_i$ is the dot product. Note that we define the logistic regression function to output probabilities, not data labels. Let $\texttt{LOGIT}_n$ denote the class of logistic regression functions with n variables, and $\texttt{LOGIT} = \bigcup_n \texttt{LOGIT}_n$. We prove the following:

Theorem 5. *Computing the expectation of a logistic regression model w.r.t. a uniform data distribution is #P-hard.*

The full proof in Appendix C is by reduction from counting solutions to the number partitioning problem.

Because the uniform distribution is contained in $\texttt{IND}$, and following Theorem 1, we immediately obtain:

[1] In contemporaneous work, Arenas et al. (2020) also show that the SHAP explanation is tractable for d-DNNFs, but for the more restricted class of uniform data distributions.

Corollary 6. *The computational problems* $\mathtt{E(LOGIT)}$ *and* $\mathtt{F\text{-}SHAP(LOGIT,IND)}$ *are both #P-hard.*

We are now ready to list general function classes for which computing the SHAP explanation is #P-hard.

Corollary 7. *For the following function classes* $\mathtt{F}$*, computing* SHAP *scores* $\mathtt{F\text{-}SHAP(F,IND)}$ *is #P-hard in the size of the representations of function* $F \in \mathtt{F}$ *and fully-factorized distribution* $\Pr \in \mathtt{IND}$.

1. *Logistic regression models (Corollary 6)*
2. *Neural networks with sigmoid activation functions*
3. *Naive Bayes classifiers, logistic circuits*
4. *Boolean functions in CNF or DNF*

Our intractability results stem from these observations:

2. Each neuron is a logistic regression model, and therefore this class subsumes $\mathtt{LOGIT}$.
3. The conditional distribution used by a naive Bayes classifier is known to be equivalent to a logistic regression model (Ng and Jordan 2002). Logistic circuits are a graph-based classification model that subsumes logistic regression (Liang and Van den Broeck 2019).
4. For general CNFs and DNFs, weighted model counting, and therefore $\mathtt{E(F)}$ is #P-hard. This is true even for very restricted classes, such as monotone 2CNF and 2DNF functions, and Horn clause logic (Wei and Selman 2005).

4 Beyond Fully-Factorized Distributions

Features in real-world data distributions are not independent. In order to capture more realistic assumptions about the data when computing SHAP scores, one needs a more intricate probabilistic model. In this section we prove that the complexity of computing the SHAP-explanation quickly becomes intractable, even over the simplest probabilistic models, namely naive Bayes models. To make computing the SHAP-explanation as easy as possible, we will assume that the function F simply outputs the value of one feature. We show that even in this case, even for function classes that are tractable under fully-factorized distributions, computing SHAP explanations becomes computationally hard.

Let $\mathtt{NBN}_n$ denote the family of naive Bayes networks over $n+1$ variables $\mathbf{X} = \{X_0, X_1, \dots, X_n\}$, with binary domains, where X_0 is a parent of all features:

$$\Pr(\mathbf{X}) = \Pr(X_0) \cdot \prod_{i=1,n} \Pr(X_i | X_0).$$

As usual, the class $\mathtt{NBN} \overset{\text{def}}{=} \bigcup_{n \geq 0} \mathtt{NBN}_n$. We write X_0 for the function F that returns the value of feature X_0; that is, $F(\mathbf{x}) = \mathbf{x}_0$. We prove the following.

Theorem 8. *The decision problem* $\mathtt{D\text{-}SHAP}(\{X_0\}, \mathtt{NBN})$ *is NP-hard.*

The proof in Appendix D is by reduction from the number partitioning problem, similar to the proof of Corollary 6. We note that the subset sum problem was also used to prove related hardness results, e.g., for proving hardness of the Shapely value in network games (Elkind et al. 2008).

This result is in sharp contrast with the complexity of the SHAP score over fully-factorized distributions in Section 3. There, the complexity was dictated by the choice of function class $\mathtt{F}$. Here, the function is as simple as possible, yet computing SHAP is hard. This ruins any hope of achieving tractability by restricting the function, and this motivates us to restrict the probability distribution in the next section. This result is also surprising because it is efficient to compute marginal probabilities (such as the expectation of X_0) and conditional probabilities in naive Bayes distributions.

Theorem 8 immediately extends to a large class of probability distributions and functions. We say that F *depends only on* X_i if there exist two constants $c_0 \neq c_1$ such that $F(\mathbf{x}) = c_0$ when $\mathbf{x}_i = 0$ and $F(\mathbf{x}) = c_1$ when $\mathbf{x}_i = 1$. In other words, F ignores all variables other than X_i, and *does* depend on X_i. We then have the following.

Corollary 9. *The problem* $\mathtt{D\text{-}SHAP(F,PR)}$ *is NP-hard, when* $\mathtt{PR}$ *is any of the following classes of distributions:*

1. *Naive Bayes, bounded-treewidth Bayesian networks*
2. *Bayesian networks Markov networks, Factor graphs*
3. *Decomposable probabilistic circuits*

and when $\mathtt{F}$ *is any class that contains some function* F *that depends only on* X_0*, including the class of linear regression models and all the classes listed in Corollaries 4 and 7.*

This corollary follows from two simple observations. First, each of the classes of probability distributions listed in the corollary can represent a naive Bayes network over binary variables $\mathbf{X}$. For example, a Markov network will consists of n factors $f_1(X_0, X_1), f_2(X_0, X_2), \dots, f_n(X_0, X_n)$; similar simple arguments prove that all the other classes can represent naive Bayes, including tractable probabilistic circuits such as sum-product networks (Vergari et al. 2020).

Second, for each function that depends only on X_0, there exist two distinct constants $c_0 \neq c_1 \in \mathbb{R}$ such that $F(\mathbf{x}) = c_0$ when $\mathbf{x}_0 = 0$ and $F(\mathbf{x}) = c_1$ when $\mathbf{x}_0 = 1$. For example, if we consider the class of logistic regression functions $F(\mathbf{x}) = \sigma(\sum_i w_i \mathbf{x}_i)$, then we choose the weights $w_0 = 1$, $w_1 = \dots = w_n = 0$ and we obtain $F(\mathbf{x}) = 1/2$ when $\mathbf{x}_0 = 0$ and $F(\mathbf{x}) = 1/(1+e^{-1})$ when $\mathbf{x}_0 = 1$. Then, over the binary domain $\{0,1\}$ the function is equivalent to $F(\mathbf{x}) = (c_1 - c_0)\mathbf{x}_0 + c_0$, and, therefore, by the linearity of the SHAP explanation (Equation 4) we have $\textsc{Shap}_F(X_0) = (c_1 - c_0) \cdot \textsc{Shap}_{X_0}(X_0)$ (because the SHAP explanation of a constant function c_0 is 0) for which, by Theorem 8, the decision problem is NP-hard.

We end this section by proving that Theorem 8 continues to hold even if the prediction function F is the value of some leaf node of a (bounded treewidth) Bayesian Network. In other words, the hardness of the SHAP explanation is not tied to the function returning the root of the network, and applies to more general functions.

Corollary 10. *The* SHAP *decision problem for Bayesian networks with latent variables is NP-hard, even if the function* F *returns a single leaf variable of the network.*

The full proof is given in Appendix E.

5 Shap on Empirical Distributions

In supervised learning one does not require a generative model of the data, instead, the model is trained on some concrete data set: the *training data*. When some probabilistic model is needed, then the training data itself is conveniently used as a probability model, called the *empirical distribution*. This distribution captures dependencies between features, while its set of possible worlds is limited to those in the data set. For example, the intent of the KernelSHAP algorithm by Lundberg and Lee (2017) is to compute the Shap explanation on the empirical distribution. In another example, Aas, Jullum, and Løland (2019) extend KernelSHAP to work with dependent features, by estimating the conditional probabilities from the empirical distribution.

Compared to the data distributions considered in the previous sections, the empirical distribution has one key advantage: it has many fewer possible worlds with positive probability – this suggests increased tractability. Unfortunately, in this section, we prove that computing the Shap explanation over the empirical distribution is #P-hard in general.

To simplify the presentation, this section assumes that all features are binary: $\mathrm{dom}(X_j) = \{0, 1\}$. The probability distribution is given by a 0/1-matrix $\boldsymbol{d} = (x_{ij})_{i\in[m],j\in[n]}$, where each row $(x_{i1}, \ldots, x_{in})$ is an outcome with probability $1/m$. One can think of $\boldsymbol{d}$ as a dataset with n features and m data instances, where each row $(x_{i1}, \ldots, x_{in})$ is one data instance. Repeated rows are possible: if a row occurs k times, then its probability is k/m. We denote by $\mathtt{X}$ the class of empirical distributions. The predictive function can be any function $F : \{0, 1\}^n \to \mathbb{R}$. As our data distribution is no longer strictly positive, we adopt the standard convention that $\mathbf{E}[F|\mathbf{X}_S = 1] = 0$ when $\Pr(\mathbf{X}_S = 1) = 0$.

Recall from Section 2.2 that, by convention, we compute the Shap-explanation w.r.t. instance $\mathbf{e} = (1, 1, \ldots, 1)$, which is without loss of generality. Somewhat surprisingly, the complexity of computing the Shap-explanation of a function F over the empirical distribution given by a matrix $\boldsymbol{d}$ is related to the problem of computing the expectation of a certain CNF formula associated to $\boldsymbol{d}$.

Definition 4. The *positive, partitioned 2CNF formula*, PP2CNF, associated to a matrix $\boldsymbol{d} \in \{0, 1\}^{m\times n}$ is:

$$\Phi_{\boldsymbol{d}} \stackrel{\text{def}}{=} \bigwedge_{(i,j):x_{ij}=0} (U_i \vee V_j).$$

Thus, a PP2CNF formula is over $m + n$ variables $U_1, \ldots, U_m, V_1, \ldots, V_n$, and has only positive clauses. The matrix $\boldsymbol{d}$ dictates which clauses are present. A *quasy-symmetric probability distribution* is a fully factorized distribution over the $m + n$ variables for which there exists two numbers $p, q \in [0, 1]$ such that for every $i = 1, m$, $\Pr(U_i = 1) = p$ or $\Pr(U_i = 1) = 1$, and for every $j = 1, n$, $\Pr(V_j = 1) = q$ or $\Pr(V_j = 1)$. In other words, all variables $U_1, \ldots, U_m$ have the same probability p, or have probability 1, and similarly for the variables $V_1, \ldots, V_n$. We denote by $\mathtt{EQS(PP2CNF)}$ the expectation computation problem for PP2CNF over quasi-symmetric probability distributions. $\mathtt{EQS(PP2CNF)}$ is #P-hard, because computing $\mathbf{E}[\Phi_{\boldsymbol{d}}]$ under the uniform distribution (i.e. $\Pr(U_1 = 1) = \cdots = \Pr(V_n = 1) = 1/2$) is #P-hard (Provan and Ball 1983). We prove:

Theorem 11. *Let* $\mathtt{X}$ *be the class of empirical distributions, and* $\mathtt{F}$ *be any class of function such that, for each* i*, it includes some function that depends only on* X_i*. Then, we have that* $\mathtt{F\text{-}SHAP(F, X)} \equiv^P \mathtt{EQS(PP2CNF)}$*.*

As a consequence, the problem $\mathtt{F\text{-}SHAP(F, X)}$ *is #P-hard in the size of the empirical distribution.*

The theorem is surprising, because the set of possible outcomes of an empirical distribution is small. This is unlike all the distributions discussed earlier, for example those mentioned in Corollary 9, which have 2^n possible outcomes, where n is the number of features. In particular, given an empirical distribution $\boldsymbol{d}$, one can compute the expectation $\mathbf{E}[F]$ in polynomial time for any function F, by doing just one iteration over the data. Yet, computing the Shap explanation of F is #P-hard.

Theorem 11 implies hardness of Shap explanations on the empirical distribution for a large class of functions.

Corollary 12. *The problem* $\mathtt{F\text{-}SHAP(F, X)}$ *is #P-hard, when* $\mathtt{X}$ *is the class of empirical distributions, and* $\mathtt{F}$ *is any class such that for each feature* X_i*, the class contains some function that depends only on* X_i*. This includes all the function classes listed in Corollaries 4 and 7.*

For instance, any class of Boolean function that contains the n single-variable functions $F \stackrel{\text{def}}{=} X_i$, for $i = 1, n$, fall under this corollary. Section 4 showed an example of how the class of logistic regression functions fall under this corollary as well.

The proof of Theorem 11 follows from the following technical lemma, which is of independent interest:

Lemma 13. *We have that:*

1. *For every matrix* $\boldsymbol{d}$*,* $\mathtt{F\text{-}SHAP}(\mathtt{F}, \boldsymbol{d}) \leq^P \mathtt{EQS}(\{\Phi_{\boldsymbol{d}}\})$*.*
2. $\mathtt{EQS(PP2CNF)} \leq^P \mathtt{F\text{-}SHAP(F, X)}$*.*

The proof of the Lemma is given in Appendix F and G. The first item says that we can compute the Shap-explanation in polynomial time using an oracle for computing $\mathbf{E}[\Phi_{\boldsymbol{d}}]$ over quasi-symmetric distributions. The oracle is called only on the PP2CNF $\Phi_{\boldsymbol{d}}$ associated to the data $\boldsymbol{d}$, but may perform repeated calls, with different probabilities of the Boolean variables. This is somewhat surprising because the Shap explanation is over an empirical distribution, while $\mathbf{E}[\Phi_{\boldsymbol{d}}]$ is taken over a fully-factorized distribution; there is no connection between these two distributions. This item immediately implies $\mathtt{F\text{-}SHAP(F, X)} \leq^P \mathtt{EQS(PP2CNF)}$, where $\mathtt{X}$ is the class of empirical distributions $\boldsymbol{d}$, since the formula $\Phi_{\boldsymbol{d}}$ is in the class $\mathtt{PP2CNF}$.

The second item says that a weak form of converse also holds. It states that we can compute in polynomial time the expectation $\mathbf{E}[\Phi]$ over a quasi-symmetric probability distributions by using an oracle for computing Shap explanations, over several matrices $\boldsymbol{d}$, but not necessarily restricted to the matrix associated to Φ. Together, the two items of the lemma prove Theorem 11.

We end this section with a comment on the TreeSHAP algorithm in Lundberg et al. (2020), which is computed over a distribution defined by a tree-based model. Our result implies that the problem that TreeSHAP tries to solve is #P-hard. This follows immediately by observing that every empirical distribution $\boldsymbol{d}$ can be represented by a binary tree of size polynomial in the size of $\boldsymbol{d}$. The tree examines the attributes in the order $X_1, X_2, \ldots, X_n$, and each decision node for X_i has two branches: $X_i = 0$ and $X_i = 1$. A branch that does not exists in the matrix $\boldsymbol{d}$ will end in a leaf with label 0. A complete branch that corresponds to a row $x_{i1}, x_{i2}, \ldots, x_{in}$ in $\boldsymbol{d}$ ends in a leaf with label $1/m$ (or k/m if that row occurs k times in $\boldsymbol{d}$). The size of this tree is no larger than twice the size of the matrix (because of the extra dead end branches). This concludes our study of SHAP explanations on the empirical distribution.

6 Perspectives and Conclusions

We establish the complexity of computing the SHAP explanation in three important settings. First, we consider fully-factorized data distributions and show that for any prediction model, the complexity of computing the SHAP explanation is the same as the complexity of computing the expected value of the model. It follows that there are commonly used models, such as logistic regression, for which computing SHAP explanations is intractable. Going beyond fully-factorized distributions, we show that computing SHAP explanations is also intractable for simple functions and simple distributions – naive Bayes and empirical distributions.

The recent literature on SHAP explanations predominantly studies tradeoffs of variants of the original SHAP formulation, and relies on approximation algorithms to compute the explanations. These approximation algorithms, however, tend to make simplifying assumptions which can lead to counter-intuitive explanations, see e.g., Slack et al. (2020). We believe that more focus should be given to the computational complexity of SHAP explanations. In particular, which classes of machine learning models can be explained efficiently using the SHAP scores? Our results show that, under the assumption of fully-factorized data distributions, there are classes of models for which the SHAP explanations can be computed in polynomial time. In future work, we plan to explore if there are classes of models for which the complexity of the SHAP explanations is tractable under more complex data distributions, such as the ones defined by tractable probabilistic circuits (Vergari et al. 2020) or tractable symmetric probability spaces (Van den Broeck, Meert, and Darwiche 2014; Beame et al. 2015).

Acknowledgements

This work is partially supported by NSF grants IIS-1907997, IIS-1954222 IIS-1943641, IIS-1956441, CCF-1837129, DARPA grant N66001-17-2-4032, a Sloan Fellowship, and gifts by Intel and Facebook research. Schleich is supported by a RelationalAI fellowship. The authors would like to thank YooJung Choi for valuable discussions on the proof of Theorem 5.

References

Aas, K.; Jullum, M.; and Løland, A. 2019. Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *arXiv preprint arXiv:1903.10464*.

Arenas, M.; Barceló, P.; Bertossi, L.; and Monet, M. 2020. The Tractability of SHAP-Score-Based Explanations over Deterministic and Decomposable Boolean Circuits. *arXiv preprint arXiv:2007.14045*.

Beame, P.; Van den Broeck, G.; Gribkoff, E.; and Suciu, D. 2015. Symmetric Weighted First-Order Model Counting. In *Proceedings of the 34th ACM Symposium on Principles of Database Systems, PODS 2015, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, 313–328.

Bertossi, L.; Li, J.; Schleich, M.; Suciu, D.; and Vagena, Z. 2020. Causality-Based Explanation of Classification Outcomes. In *Proceedings of the Fourth International Workshop on Data Management for End-to-End Machine Learning*, DEEM'20. New York, NY, USA: Association for Computing Machinery.

Bryant, R. E. 1986. Graph-based algorithms for boolean function manipulation. *Computers, IEEE Transactions on* 100(8): 677–691.

Chavira, M.; and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6-7): 772–799.

Darwiche, A.; and Marquis, P. 2002. A knowledge compilation map. *Journal of Artificial Intelligence Research* 17: 229–264.

Datta, A.; Sen, S.; and Zick, Y. 2016. Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, 598–617.

Elkind, E.; Goldberg, L. A.; Goldberg, P. W.; and Wooldridge, M. J. 2008. A tractable and expressive class of marginal contribution nets and its applications. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12-16, 2008, Volume 2*, 1007–1014.

Ferrara, A.; Pan, G.; and Vardi, M. Y. 2005. Treewidth in verification: Local vs. global. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, 489–503. Springer.

Gade, K.; Geyik, S. C.; Kenthapadi, K.; Mithal, V.; and Taly, A. 2019. Explainable AI in Industry. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 3203–3204. New York, NY, USA: Association for Computing Machinery.

Janzing, D.; Minorics, L.; and Bloebaum, P. 2020. Feature relevance quantification in explainable AI: A causal problem. volume 108 of *Proceedings of Machine Learning Research*, 2907–2916. PMLR.

Khosravi, P.; Choi, Y.; Liang, Y.; Vergari, A.; and Van den Broeck, G. 2019a. On Tractable Computation of Expected

Predictions. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*.

Khosravi, P.; Liang, Y.; Choi, Y.; and den Broeck, G. V. 2019b. What to Expect of Classifiers? Reasoning about Logistic Regression with Missing Features. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 2716–2724.

Khosravi, P.; Vergari, A.; Choi, Y.; Liang, Y.; and Van den Broeck, G. 2020. Handling Missing Data in Decision Trees: A Probabilistic Approach. In *The Art of Learning with Missing Values Workshop at ICML (Artemiss)*.

Kumar, I. E.; Venkatasubramanian, S.; Scheidegger, C.; and Friedler, S. 2020. Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the 37th International Conference on Machine Learning*, Vienna, Austria, PMLR 119, 2020.

Liang, Y.; and Van den Broeck, G. 2019. Learning Logistic Circuits. In *Proceedings of the 33rd Conference on Artificial Intelligence (AAAI)*.

Lundberg, S. M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J. M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; and Lee, S. 2020. From Local Explanations to Global Understanding with Explainable AI for Trees. *Nature Machine Intelligence* 2: 56–67.

Lundberg, S. M.; Erion, G. G.; and Lee, S.-I. 2018. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888* .

Lundberg, S. M.; and Lee, S. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in neural information processing systems (NIPS)*, 4765–4774.

Merrick, L.; and Taly, A. 2020. The Explanation Game: Explaining Machine Learning Models Using Shapley Values. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, 17–38. Springer.

Ng, A. Y.; and Jordan, M. I. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, 841–848.

Provan, J. S.; and Ball, M. O. 1983. The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. *SIAM J. Comput.* 12(4): 777–788.

Rendle, S. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*, 995–1000. IEEE.

Roth, A. e. 1988. *The Shapley Value: Essays in Honor of Lloyd S. Shapley*. Cambridge Univ. Press.

Sang, T.; Beame, P.; and Kautz, H. A. 2005. Performing Bayesian inference by weighted model counting. In *AAAI*, volume 5, 475–481.

Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; and Lakkaraju, H. 2020. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *AAAI/ACM Conference on AI, Ethics, and Society (AIES)*.

Štrumbelj, E.; and Kononenko, I. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems* 41(3): 647–665.

Sundararajan, M.; and Najmi, A. 2020. The many Shapley values for model explanation. In *Proceedings of the 37th International Conference on Machine Learning*, Vienna, Austria, PMLR 119, 2020.

Van den Broeck, G.; Meert, W.; and Darwiche, A. 2014. Skolemization for Weighted First-Order Model Counting. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*.

Vergari, A.; Choi, Y.; Peharz, R.; and Van den Broeck, G. 2020. Probabilistic Circuits: Representations, Inference, Learning and Applications. AAAI Tutorial.

Wei, W.; and Selman, B. 2005. A new approach to model counting. In *International Conference on Theory and Applications of Satisfiability Testing*, 324–339. Springer.