An Active Approach for Model Interpretation

Jialin Lu

1 Introduction

Model interpretation, or explanation of a machine learning classifier, aims to extract generalizable knowledge from a trained classifier into a human-understoodable format, for various purposes such as model assessment, debugging and trust. From a computaional viewpoint, it is formulated as approximating the target classifier using a simpler interpretable model, such as rule models like decision set/list/tree. Often and straightforwardly, this approximation is handled as standard supervised learning and the only difference is that the labels are provided by the target classifier instead of ground truth. This paradigm is particularly popular because there exists a variety of well-studied supervised algorithms for learning an interpretable classifier.

However, we argue that this paradigm is suboptimal because it does not utilize the unique property of the model interpretation problem, that is, we can generate synthetic instances that enable better approximation and query the target classifier model to get their label. We call this the *active-query* property, suggesting that we should consider model interpretation from an active learning perspective.

Following this insight, we argue that the *active-query* property should be employed when designing a model interpretation algorithm, and the generation of synthetic instances should be integrated seamlessly with the algorithm that learns the model interpretation. In this paper, we demonstrate that by doing so, it is possible to achieve more faithful interpretation with simpler model complexity.

As a technical contribution, we present an active algorithm, called **AlgoXXX** , to learn a decision set, a set of if-else rules, for model interpretation. **AlgoXXX** performs a local search of the space of all decision sets. In every iteration, **AlgoXXX** computes confidence intervals for all candidate local actions and utilizes active-query to determine the best one to apply.

We believe that the active query paradigm will be beneficial also for the design of other active algorithms for model interpretation, especially iteration-based discrete optimization algorithms.

2 Related work

We discuss global model interpretation methods that approximates the global behavior of a target classifier and we further restrict ourselves to rule-based methods.

We first introduce the mainstream of related work which we refer as *passive* model interpretation, in the sense that it approximates a fixed-sized given dataset labelled by the target classifier, rather than the target classifier itself. Lakkaraju et al. (2017, 2019) learns a decision set for model interpretation by jointly optimizing interpretability and approximation faithfulness on the relabelled dataset. It is widely cited and there are many alternatives for learning an interpretable classifier (Lakkaraju et al., 2016; Wang et al., 2017; Yang et al., 2017). But the problem is that the given dataset, in practice, may not be good enough to reveal the true decision boundary and thus the approximation faithfulness is not guaranteed. Following approaches (Ming et al., 2018; Paçaci et al., 2019; Sushil et al., 2018) first augment the dataset and then apply the main algorithm for learning an interpretable model from this dataset, for example, Ming et al. (2018) first estimate $P(X)$ and then sample an arbitray large dataset. We emphasize that *passive* methods could utilize new data instances, but only

Preprint. Under review.

as a preprocessing procedure which is separate from the main learning algorithm, and thus we argue is sub-optimal.

Another line of work, which we call *bottom-up* methods, applys two-phase local-to-global algorithms (Pedreschi et al., 2018) leveraging the relatively well-studied local interpretation methods (Ribeiro et al., 2016, 2018; Guidotti et al., 2018). The idea is to first construct an local interpretation for each instance in the dataset, and then select a subset of them to constitute a global interpretation.

Lastly, we introduce the closest work to ours, DT-Extract (Bastani et al., 2017), as it integrates active-query seamlessly into the algorithm. It modifies the classic decision tree algorithm CART by generating new instances at each node to select a best split and thus better approximates the target classifier. We believe this tree-growing approach has been re-invented many times (Bastani et al., 2017; Craven, 1996; Breiman and Shang, 1996) though at different contexts and suffers a critical disadvantage as its approximation does not consider explicitly interpretability, as an inherent drawback of decision trees. In order to be interpretable, the number of node split is hard-coded and thus the number of leave nodes is controlled. We refer to these methods as *top-down* methods.

3 Definitions and Problem Formulation

In our problem, we are given a pre-trained binary classifier f that receives an instance x and provides a prediction $f(x) \in \{0, 1\}$. Let $X = \{x\}^n = \{(a_1, a_2, ..., a_m)\}^n$ represents the instance set of n instances with m attributes. We further define the input space as $D_x = \{dom(a_i)\}^m$. We consider the form of model interpretation to be a decision set, consisting of a set of if-then rules. It predicts the positive class if at least one of the rules is satisfied and predicts the negative class if none of the rules are satisfied. A condition (clause) c, consisting of an attribute and a range of values this attribute can take. For continuous attributes, we use a bounded interval, for example, 'price' $\in [2.33, 10]$, and for categorical features we use a list of values, for example, 'state' $\in \{\textit{California}, \textit{Texas}\}$. A rule is a conjunction of conditions $r = c_1 \wedge c_2 \wedge ....$ We use $r(\cdot)$ to represent the Boolean function $r(\cdot) : D_x \mapsto \{0, 1\}$ to indicate whether or not a instance x satisfies r.

Definition 1. A decision set is a set of rules, denoted as $S = \{r_1, r_2, ...\}$. An instance is predicted as positive if it satisfies at least one of the rules, as defined by the predicting function $S(\cdot)$:

$$S(x) = \begin{cases} 1 & \exists r \in S, r(x) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

We consider to maximize a simple objective function $Q(S) = \theta_S - \lambda|S|$ where the first term $\theta_S = E_{x \sim P(X)}[\mathbb{1}_{f(x)=S(x)}]$ measures the faithfulness of the approximation of our decision set to the target classifier, defined as the expected accuracy over the input space. $|S|$ denotes the number of rules in S as a measure of interpretability. λ is a tunable weight set to 0.01 by default, meaning that an 1% accuracy improvement is worth adding an extra rule.

Remark. We do not consider the design of objective function to be the main focus of this paper. Any reasonable heuristics should be plausible such as (Lakkaraju et al., 2016, 2019; Wang et al., 2017), as interpretability comes in different forms in different domains (Freitas, 2014; Huysmans et al., 2011)

4 Active Decision Set Learning (AlgoXXX algorithm)

Given a target classifier f and a data set of real data instances X, the goal of our algorithm **AlgoXXX** is find a decision set that maximizes the objective function Q. We formulate the algorithm as a local search and utilize active query when it is necessary to determine the best action in each iteration. We initialize the decision set S as empty and in each iteration, we generate a set of actions A to modify the current decision set and greedily choose the best action $a \in A$ to improve the objective. The actions are modifications to the current decision set, such as removing a rule, or modifying a condition of a rule. After we get a set of actions, we utilize active query to determine the objective increase of each action and perform greedy hill-climbing. To escape from local optima, we employ a simple ϵ-greedy strategy: with probability ϵ, we choose a random action; otherwise we go greedy, choosing the action that improves the objective the most.

Actions Generation We consider actions as modifications on the current decision set, with a edit distance of 1: the resulted decision set after action and current set only have one rule different. We

apply actions of the following types: 1) add a rule 2) remove a rule 3) add a condition 4) remove a condition 5) modify the value of a condition.

When to utilize *active query*? In the framework of local search, we keep doing hill-climbing, choosing the best action maximizing the objective for each iteration. For a fixed-size dataset, this argmax step is trivial. However the critical challenge of model interpretation problem is how to determine the best action, because the faithfulness term $\theta_S = E_{x\sim P(x)}[\mathbb{1}_{f(x)=a(S)(x)}]$ from the objective function can only be estimated as $\hat{\theta}_S = E_{x\in X\cup X'}[\mathbb{1}_{f(x)=a(S)(x)}]$, where X' is the synthetic instances we generated. It is at this point of time for determining the best action that we utilize active query. We borrow ideas from the best arm identification problem for pure-exploration multi-arm bandits (Audibert and Bubeck, 2010; Kalyanakrishnan et al., 2012; Kaufmann and Kalyanakrishnan, 2013), particularly the LUCB algorithm (Kalyanakrishnan et al., 2012). Note that since the rules of S are descriptive, we could use the rules to generate new instance and adaptively query the classifier to better estimate $\hat{\theta}_S$, and thus determine the best action (arm). For estimating $\hat{Q}$ of applying an action a on the current decision set S, we compute a confidence interval to construct lower bound and a upper bound for $\hat{Q}$. For some exploration rate β as a hyper-parameter, the lower bound and upper bound are computed as:

$$L_a = \hat{Q}(a(S)) - \beta\sqrt{\frac{\rho_0}{\rho(r_a)}}\,, U_a = \hat{Q}(a(S)) + \beta\sqrt{\frac{\rho_0}{\rho(r_a)}}$$

β is a hyper-parameter that controlls the confidence level and $\rho(r_a) = \frac{N(r_a)}{V(r_a)}$ defines the relative density: r_a is the rule that is modified by action a , $N(r_a)$ denotes the number of instances covered by this modified rule $\{x|r_a(x) = 1\}$ and $V(r_a)$ denotes the volume of input space covered by r_a. ρ_0 is the pre-computed density of the given dataset over the entire input space.

Utilizing *active-query* exhaustively for each local action is possible, but the total number of actions can be very large, not to mention we have to do it iteratively for each iteration. We choose to use the *adaptive sampling scheme* (Kaufmann and Kalyanakrishnan, 2013), that is, we only focus on two well-chosen candidate actions: 1) the empirically best action that has the highest emprical estimation $a^* \leftarrow \underset{a\in A}{\operatorname{argmax}}\,\hat{Q}(a(S))$ and 2) the most optimistic action that has the highest upper bound $a' \leftarrow \underset{a\in A, a\neq a^*}{\operatorname{argmax}}\,\hat{Q}(a(S))$. We then generate new instances and query the target classifier to update the estimation and the bounds for these two candidate actions. This process continues until the lower bound of a^* is higher than the upper bound of a', i.e., $L_{a^*} > U_{a'}$. If this termination condition is met, then we will be confident that a^* is the best action upon all other actions $a \in A$ under the exploration rate β. We then update $S = a^*(S)$ and continue to next iteration.

How to generate instance for active query? The descriptive nature of rules gives us an opportunity to generate new instances to query in order to refine the estimation of an action. In order to distinguish $\hat{Q}(a^*(S))$ and $\hat{Q}(a'(S))$ to determine the best action, at each round as long as the termination is not met, we generate one new instance for a^* and a', respectively, and query the classifier f to get the label. The generation of new instances is tackled as pool-based sampling: simply speaking, to refine $\hat{Q(a(S))}$, we first generate a pool of candidate instances and then select one to label it. For generating a pool of synthetic instances as candidates for action a, note that different actions at an iteration has only an edit distance of 1, thus we focus on r_a, the modified rule by applying a, and generate new instance for r_a. We randomly generate a pool instances that satisfies r_a, as $X_{pool} = \{x|r_a(x) = 1\}$.

We use a simple pool-generation scheme, which we call **Counterfactual Sampling**. It randomly picks up instances from $\{x|x\ inX \cup X', r_a(x) = 0\}$, which is, instances that do not satisfy r_a, and modify them to satisfies r_a by replacing the value of the specified attributes by the condition of r_a with uniform-randomly sampled values. Here we give roadmap and reasons for coming up with this idea. To sample $X_{pool} = \{x|r_a(x) = 1\}$, We first try uniformly randomness but it does not work well. We think it is because that it generates instances that is not from the data distribution $P(X)$. We then try to first estimate $P(X)$ and then do rejection-sampling $\{x|x \sim P(X), r_a(x) = 1\}$, as suggest by Ming et al. (2018). But it only improves slightly, perhaps because estimating $P(X)$ is hard and the quality of estimating $P(X)$ matters. The proposed Counterfactual sampling avoids estimating the true data distribution $P(X)$ and still works well to generate instances *as if* they were from $P(X)$.

For selecting one to label, we choose one synthetic instance out of the candidate pool that maximizes the distance to its nearest neighbor to all existing instances satisfying r_a ($\{x|r_a(x) = 1, x \in X \cup X'\}$). We then query f to get the label and append it to X' to refine $\hat{Q}(a(S))$.

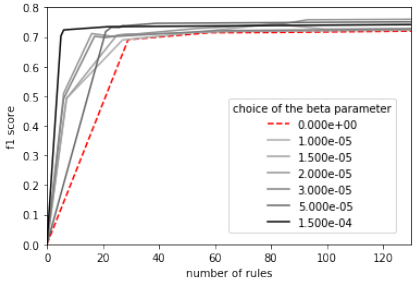
Using the largest distance to its nearest neighbor has been the simplest form of uncertainty-based active sampling heuristic. While sophisticated heuristics to measure uncertainty exists (MacKay, 1992; Bryan et al., 2006), we choose to use this simple criteria, not only because that most of uncertainty measures are essentially based on the distance to nearest neighbors, but also we find that it works pracically very well and is computationally efficient.

5 Evaluation and Conclusion

We choose to evaluate on the Income Census Prediction dataset (Kohavi, 1996), because is is a relatively large tabular dataset (48842 instances) and it is publicly available for easy reproduction. The dataset is a binary classification task predicting whether a person makes over 50K a year. We use a 90%-10% train-test split train a 5-layer Deep neural network on the training set. For baselines, we consider one representative method for each of three category of discussed related work. For passive method, we use Ming et al. (2018) which first enlarges the dataset and then learns an interpretable classifier. We change the model from a rule list (Yang et al., 2017) to a decision set (Bayesian Rule Set (Wang et al., 2017)). We also tried some other decision set methods (Lakkaraju et al., 2016), but find them not as good as Bayesian Rule Set (Wang et al., 2017). For *bottom-up* methods that constitute local interpretations into a global one, we use SP-anchor (Ribeiro et al., 2018). We also compare DT-Extract (Bastani et al., 2017) as a representative of top-down active methods, we extract all path-to-leave-node as decision rules out of the grown tree.

We show that utilizing active-query helps to find better approximation with simpler model complexity (see Figure 1), in terms of the number of rules, which is a very positive sign of active-query paradigm. We also show that our method **AlgoXXX** outperforms the baselines (see Table 1)

AlgoXXX is an active approach for model interpretation. Though very simple, it demonstrates that the active-query paradigm is highly potential.



We run our method with different choice of the exploration rate β, a larger value of β is going to use more synthetic instances and when $\beta = 0$ the algorithm will reduce to a passive method that does not use any new instances rather than the given dataset. For each given β, different choice of λ results in decision sets of different number of rules. A large λ enforces a smaller number of rules and a small λ could achieve higher faithfulness with more rules.

Figure 1: *Tuning the parameter λ and β*

Faifulness metric	ours active	ours non active	BRS+	SP-Anchor	DTExtract
acc	0.897	0.884	0.882	0.844	0.850
f1 score	0.741	0.690	0.709	0.471	0.496
recall	0.688	0.602	0.666	0.325	0.343
precision	0.803	0.809	0.757	0.860	0.8975
Interpretability metric	ours active	ours non active	BRS	SP-Anchor	DTExtract
number of rules	24	29	57	100	45
average number of conditions	4.375	2.413	3.859	7.34	6
max number of conditions	6	4	5	12	9

Table 1: Comparision of methods in the Income Census Prediction task, on various metrics of faifulness and interpretability

References

Jean-Yves Audibert and Sébastien Bubeck. 2010. Best arm identification in multi-armed bandits.

Osbert Bastani, Carolyn Kim, and Hamsa Bastani. 2017. Interpreting Blackbox Models via Model Extraction. *CoRR* abs/1705.08504 (2017). arXiv:1705.08504 http://arxiv.org/abs/1705.08504

Leo Breiman and Nong Shang. 1996. Born again trees. (1996).

Brent Bryan, Robert C. Nichol, Christopher R Genovese, Jeff Schneider, Christopher J. Miller, and Larry Wasserman. 2006. Active Learning For Identifying Function Threshold Boundaries. In *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. C. Platt (Eds.). MIT Press, 163–170. http://papers.nips.cc/paper/2940-active-learning-for-identifying-function-threshold-boundaries.pdf

Mark William Craven. 1996. *Extracting Comprehensible Models from Trained Neural Networks*. Ph.D. Dissertation. AAI9700774.

Alex A. Freitas. 2014. Comprehensible classification models. *ACM SIGKDD Explorations Newsletter* 15, 1 (mar 2014), 1–10. https://doi.org/10.1145/2594473.2594475

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local Rule-Based Explanations of Black Box Decision Systems. *CoRR* abs/1805.10820 (2018). arXiv:1805.10820 http://arxiv.org/abs/1805.10820

Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. 2011. An empirical evaluation of the comprehensibility of decision table tree and rule based predictive models. *Decision Support Systems* 51, 1 (apr 2011), 141–154. https://doi.org/10.1016/j.dss.2010.12.003

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. 2012. PAC Subset Selection in Stochastic Multi-armed Bandits.. In *ICML*, Vol. 12. 655–662.

Emilie Kaufmann and Shivaram Kalyanakrishnan. 2013. Information complexity in bandit subset selection. In *Conference on Learning Theory*. 228–251.

Ron Kohavi. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-tree Hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 202–207. http://dl.acm.org/citation.cfm?id=3001460.3001502

Himabindu Lakkaraju, Stephen H. Bach, and Jure Leskovec. 2016. Interpretable Decision Sets: A Joint Framework for Description and Prediction. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 1675–1684. https://doi.org/10.1145/2939672.2939874

Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. 2017. Interpretable & Explorable Approximations of Black Box Models. *CoRR* abs/1707.01154 (2017). arXiv:1707.01154 http://arxiv.org/abs/1707.01154

Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. 2019. Faithful and Customizable Explanations of Black Box Models. *AIES* (2019).

David J. C. MacKay. 1992. Information-Based Objective Functions for Active Data Selection. *Neural Computation* 4, 4 (jul 1992), 590–604. https://doi.org/10.1162/neco.1992.4.4.590

Yao Ming, Huamin Qu, and Enrico Bertini. 2018. RuleMatrix: Visualizing and Understanding Classifiers with Rules. *CoRR* abs/1807.06228 (2018). arXiv:1807.06228 http://arxiv.org/abs/1807.06228

Görkem Paçaci, David Johnson, Steve McKeever, and Andreas Hamfelt. 2019. Why did you do that?: Explaining black box models with Inductive Synthesis. *CoRR* abs/1904.09273 (2019). arXiv:1904.09273 http://arxiv.org/abs/1904.09273

Dino Pedreschi, Fosca Giannotti, Riccardo Guidotti, Anna Monreale, Luca Pappalardo, Salvatore Ruggieri, and Franco Turini. 2018. Open the Black Box Data-Driven Explanation of Black Box Decision Systems. *CoRR* abs/1806.09936 (2018). arXiv:1806.09936 http://arxiv.org/abs/1806.09936

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why Should I Trust You?. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 16*. ACM Press. https://doi.org/10.1145/2939672.2939778

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*.

Madhumita Sushil, Simon Suster, and Walter Daelemans. 2018. Rule induction for global explanation of trained models. *CoRR* abs/1808.09744 (2018). arXiv:1808.09744 http://arxiv.org/abs/1808.09744

Tong Wang, Cynthia Rudin, Finale Doshi-Velez, Yimin Liu, Erica Klampfl, and Perry MacNeille. 2017. A Bayesian Framework for Learning Rule Sets for Interpretable Classification. *Journal of Machine Learning Research* 18, 70 (2017), 1–37. http://jmlr.org/papers/v18/16-003.html

Hongyu Yang, Cynthia Rudin, and Margo Seltzer. 2017. Scalable Bayesian Rule Lists. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70 (ICML'17)*. JMLR.org, 3921–3930. http://dl.acm.org/citation.cfm?id=3305890.3306086