Learning Global Transparent Models from Local Contrastive Explanations

Tejaswini Pedapati, Avinash Balakrishnan, Karthikeyan Shanmugam, and Amit Dhurandhar

IBM Research

Abstract

There is a rich and growing literature on producing local point wise contrastive/counterfactual explanations for complex models. These methods highlight what is important to justify the classification and/or produce a contrast point that alters the final classification. Other works try to build globally interpretable models like decision trees and rule lists directly by efficient model search using the data or by transferring information from a complex model using distillation-like methods. Although these interpretable global models can be useful, they may not be consistent with local explanations from a specific complex model of choice. In this work, we explore the question: Can we produce a transparent global model that is consistent with/derivable from local explanations? Based on a key insight we provide a novel method where every local contrastive/counterfactual explanation can be turned into a Boolean feature. These Boolean features are sparse conjunctions of binarized features. The dataset thus constructed is consistent with local explanations by design and one can train an interpretable model like a decision tree on it. We note that this approach strictly loses information due to reliance only on sparse local explanations, nonetheless, we demonstrate empirically that in many cases it can still be competitive with respect to the complex model's performance and also other methods that learn directly from the original dataset. Our approach also provides an avenue to benchmark local explanation methods in a quantitative manner.

1 Introduction

With the ever increasing adoption of black-box artificial intelligence technologies in various facets of society [9], a lot of interpretable algorithms have been used to explain the decisions taken by the black box models. They are chiefly of two types: a) Local Explanations for data point of interest [24, 12, 13, 14, 19, 6] and b) Constructing interpretable global models directly like decision trees, rule lists and boolean rules [16]. One of the arguments for b) is that it is sometimes

possible to construct interpretable global models in such a way that for a single data point it can give a succinct local explanation in the form of a sparse conjunction [16]. Methods in a) naturally enjoy the parsimonious explanations on a single data point through either feature importance scores or contrastive points that differ in very few features. Another option is to derive a global interpretable model out of a complex one through means of information transfer [8, 7]. However, these new models may not be locally faithful to the original explanations of a complex black-box model.

As such, there has been limited amount of effort in directly leveraging the sparse local explanations which are *logically conjunction of conditions on few features* to build transparent models without sacrificing too much in performance. Potentially, an approach of building a globally transparent model from local explanations would retain the structure (sparse interaction of various features) of the local explanations based on a complex black-box model that is preferred in some specific application (Boosted Trees or Neural Nets being preferable to deploy due to accuracy).

We propose a new algorithm that uses local explanations from a contrastive explanations method to generate boolean clauses which are conjunctions. These boolean conjunctions can be used as features, forming a new dataset, to train another simple model like a sparse linear model such as logistic regression or a small decision tree. The algorithm binarizes local contrastive explanations depending on the difference in feature values between the contrast point and the original. The binarization of features is directed by local explanations based on ranges that are deemed locally important by the complex model. One of the most interesting aspects of this idea is that sparse interactions between original features required for explaining, are directly captured by these boolean clauses.

To showcase our idea we use the model agnostic contrastive explanations method [6], that generates local explanations in the form of pertinent positives (PPs) and pertinent negatives (PNs). PPs are a minimal set of features with minimal value that are sufficient to obtain the classification of the original input. For example, given an image of a 3 say in MNIST, the PP will be some subset of non-zero intensity pixels in the 3 with corresponding grey scale values not exceeding those in the original image which has the same classification as the original image of the 3. A PN on the other is the minimal set of features that if increased will change the classification of the original image. So in our image of a 3 example, a small set of pixels say at the right top which were zero before now have positive intensity values and are perceived as horizontal line making the image look like a 5 also to the classifier would constitute a PN.

The key insight is that each explanation with our mapping can be viewed as a conjunction (or ANDing) of literals of the form $\mathbf{1}(x > p)$ (discretized PPs) and their negations (discretized PNs) forming a boolean clause. This is why we chose contrastive explanations method as our local explainability technique as we can obtain both (a sparse set of) positive as well as negative literals as opposed to a lot of explainability methods that do not generate PNs. Moreover, this formula is likely to be small since as mentioned above the method returns sparse explanations. In principle, one could take a disjunction (ORing) of all these

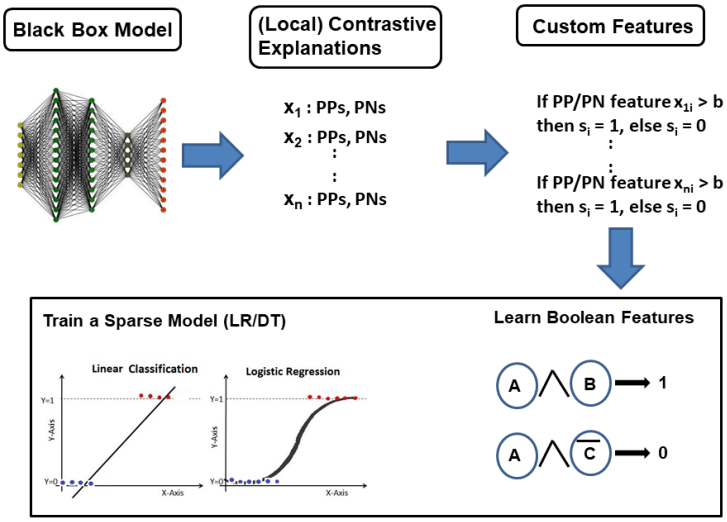

Figure 1: Above we see the high level idea behind our approach. We first obtain local contrastive explanations for the training inputs. We then transform these into boolean formulas which can serve as input to different classification models.

(local) formulas for a particular class and obtain a two-level boolean formula that can act as a global classifier. Of course, the formula would be too large and one may also have to evaluate the generalizability on a test set. Nonetheless, given these extracted formulas, which are in essence new sparse boolean conjunctions, one may be able to train a existing learner from a simple model class. We use logistic regression (with L1 penalty) or (small) decision trees as our simple base learners, however there are other possibilities (viz. boosting). These models end up consuming only a small fraction of these conjunctions. An illustration of the whole process that we just described is given in Figure 1, along with an example formula for an input in Figure 2.

2 Related Work

Most of the work on explainability in artificial intelligence can be said to fall under four major categories: Local posthoc methods, global posthoc methods, directly interpretable methods and visualization based methods.

Local Posthoc Methods: Methods under this category look to generate explanations at a per instance level for a given complex classifier that is uninterpretable. Methods in this category are either proxy model based [14, 17] or look into the internals of the model [1, 5, 24, 13]. Some of these methods also work with only black-box access [14, 6]. There are also a number of methods in this category specifically designed for images [20, 1, 18].

Global Posthoc Methods: These methods try to build an interpretable

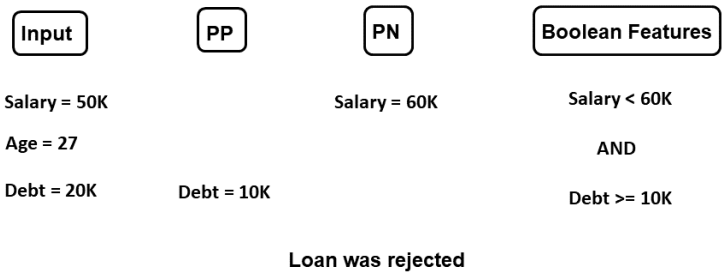

Figure 2: Above we see an example formula that we may derive for an input. A loan approval classifier has rejected the above persons loan (input features). The PP says that if the debt was reduced to 10K the loan would still have been rejected, while the PN informs us that if the salary was increased to 60K then the loan would have been accepted. Based on this we learn a rule which is shown in the last column and acts as potential feature for a model that we may train henceforth.

model on the whole dataset using information from the black-box model with the intention of approaching the black-box models performance. Methods in this category either use predictions (soft or hard) of the black-box model to train simpler interpretable models [8, 2, 3] or extract weights based on the prediction confidences reweighting the dataset [7].

Directly Interpretable Methods: Methods in this category include some of the traditional models such as decision trees or logistic regression. There has been a lot of effort recently to efficiently and accurately learn rule lists [15, 16] or two-level boolean rules [23] or decision sets [21]. There has also been work inspired by other fields such as psychometrics [11] and healthcare [4].

Visualization based Methods: These try to visualize the inner neurons or set of neurons in a layer of a neural network [10]. The idea is that by exposing such representations one may be able to gauge if the neural network is in fact capturing semantically meaningful high level features.

The most relevant categories to our current endeavor are possibly the local and global posthoc methods. The global posthoc methods although try to capture the global behavior of the black-box models, the coupling is weak as it is mainly through trying to match the output behavior, and they do not leverage or are necessarily consistent with the local explanations one might obtain.

3 Method

In this section we first describe the strategy to obtain local contrastive explanations for arbitrary black-box models. We then show how our method Global Boolean Feature Learning (GBFL) maps these explanations to boolean formulas, which is is our main contribution, that can subsequently be consumed by simple models as features to learn on.

3.1 Obtaining Contrastive Explanations

To learn our boolean features we first need a local explainability technique that can extract contrastive explanations for us from arbitrary black-box models. The method we use is the model agnostic contrastive explanations method [6], which can generate PPs and PNs with just black-box access.

Formal Definitions: We describe the definition of a PP and a PN formally. We also describe what we would get if we use the constrastive explanations method of [6]. Consider a training dataset consisting of samples $(\mathbf{x}, y) \in \mathcal{D}$. i denotes the i-th training sample. $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathcal{Y}$. $\mathcal{Y}$ is a finite set of class labels. Let us denote the training dataset by $\mathcal{D}$.

Base Value Vector: To find PPs/PNs their method requires specifying values for each feature that are least interesting, which they term as base values. A user can prespecify semantically meaningful base values or a default value of say median could be set as a base value. Classifiers essentially pick out the correlation between variation from this value in a given co-ordinate and the target class y. Therefore, we define a vector of base values $\mathbf{b} \in \mathbb{R}^d$. Variation away from the base value is used to correlate with the target class. b_j represents the base value of the j-th feature.

Upper and Lower Bounds: Let L_j and U_j be lower and upper bounds for the j-th feature x_j.

Consider a pre-trained classifier C and let $C(\mathbf{x}, y)$ denote the classifiers confidence score (a probability) of class y given $\mathbf{x}$.

Pertinent Positive: Let $\mathbf{p}(\mathbf{x}) \in \mathbb{R}^d$ denote the pertinent positive vector associated with a training sample $(\mathbf{x}, y) \in \mathcal{D}$.

$$\begin{aligned}
&\mathbf{p}(\mathbf{x}) = \arg\min_{\delta} \max\{\max_{\tilde{y} \neq y} C(\delta, \tilde{y})\} - C(\delta, y), -\kappa\} \\
&\text{s.t } L_j \leq \delta_j \leq U_j \\
&\|\delta - \mathbf{b}\|_0 \leq k. \\
&\delta_j \leq x_j,\ j : x_j \geq b_j \\
&\delta_j \geq x_j,\ j : x_j \leq b_j
\end{aligned} \tag{1}$$

In other words, Pertinent positive is a sparse vector that the classifier C classifies in the same class as the original input with high confidence. Being sparse, it is expected to have lesser variation away from the base values than $\mathbf{x}$.

Pertinent Negative: Let $\mathbf{n}(\mathbf{x}) \in \mathbb{R}^d$ denote the pertinent positive vector associated with a training sample $(\mathbf{x}, y) \in \mathcal{D}$.

$$\begin{aligned}
&\mathbf{n}(\mathbf{x}) = \arg\min_{\mathbf{x}+\delta} \max\{C(\mathbf{x} + \delta, y) - \max_{\tilde{y} \neq y}\{C(\mathbf{x} + \delta, \tilde{y})\}, -\kappa\} \\
&\text{s.t } L_j \leq \delta_j \leq U_j \\
&\|\delta\|_0 \leq k. \\
&\delta_j \geq 0,\ j : x_j \geq b_j \\
&\delta_j \leq 0,\ j : x_j \leq b_j
\end{aligned} \tag{2}$$

In other words, Pertinent negative is a vector where there are few coordinates which are different from $\mathbf{x}$. Also, it forces the classifier C to classify it in some other class with high confidence, and in coordinates where it differs from $\mathbf{x}$, those coordinates are farther from the base values than those of $\mathbf{x}$.

Remark: Although, the method in [6] does not really perform the constrained optimization in (1,2) but uses regularization like 'Elasticnet' penalty to impose sparsity, we will assume that our PPs and PNs are the result of these optimizations just for simplicity of exposition. The only difference is that the sparsity k cannot be pre-determined but is typically a constant for many training samples in practice.

3.2 Generating Sparse Boolean Clauses from Pertinent Positives and Pertinent Negatives

The following is the key observation of our work that lets us mine interpretable Boolean features.

Key Idea: For a give training point $\mathbf{x}$, we observe that a pertinent positive and negative give rise to a $2k$-sparse Boolean AND clause as follows.

Pertinent positives says simply that a specific feature has to have at least the variation of the pertinent positive feature value for it to be classified into that class. Similarly, pertinent negatives say that a specific feature can have a variation more than x_j *but not* beyond δ_j.

Therefore, one can come up with the following Boolean Clause written as product of indicators:

$$\left[\prod_{j:p[x_j]\neq b_j, p[x_j]>b_j} \mathbf{1}(z_j > p[x_j])\right] \ldots$$
$$\left[\prod_{j:p[x_j]\neq b_j,\ p[x_j]<b_j} \mathbf{1}(z_j < p[x_j])\right] \ldots$$
$$\left[\prod_{j:n[x_j]\neq x_j,\ n[x_j]>b_j} \mathbf{1}(z_j < n[x_j])\right] \ldots$$
$$\left[\prod_{j:n[x_j]\neq x_j,\ n[x_j]<b_j} \mathbf{1}(z_j > n[x_j])\right] \tag{3}$$

Bounds using grid points to regularize: In practice, these explanations are very local and hence adding further bounds will help in generalization. This is because if for some training example x, $\mathbf{1}(z_j > p[x_j])$ is the condition. Then points from some other class can also satisfy this inequality which is an infinite interval on the real line. Since these clauses are derived from local perturbations (because of ℓ_2 penalty in the optimization), they may not be valid very far from x_j.

We first need to reduce the number of distinct clauses, so only uses clauses involving grid points g_{sj} where every coordinate j has at most D grid points. We will call the matrix of $\{g_{sj}\}$ as the Grid Matrix G.

We round off all the clauses to the nearest grid points suitably and also add regularizing upper bounds using grid points that are Δ far away from the grid point involved in the clause. We describe this in detail below for 2 out of the 4 cases that arise.

For some feature x, suppose the pertinent positive p is such that $b < p < x$ where b is the base value of the feature. We find two closest grid points as follows: a) $g_i < p$ and closest to p and b) $g_j > x$ that is closest to x. Here, i and j are their indices when you sort the grid points from the lowest to the highest. Then, instead of a clause $\mathbf{1}(x > p)$, we will have conjunction of two clauses: $\mathbf{1}(x > g_i) \times \mathbf{1}(x < g_{j+\Delta})$ where Δ is a skip parameter, that we optimize over during cross validation. For a pertinent negative n for a feature satisfying $n > x > b$, we find two grid points $g_i : n > g_i > x$ and g_i is the closest to n and g_j such that $g_j < x$ and g_j is the closest to x and $j < i$. Now, instead of the clause $\mathbf{1}(x < n)$, we substitute the conjunction $\mathbf{1}(x < g_i) \times \mathbf{1}(x > g_{j-\Delta})$. Now, there are two other cases where a pertinent positive p is less than the base value and when we have a pertinent negative for an $x < b$. Similar rounding is done. We state our boolean clause generation algorithm incorporating all these ideas in Algorithm 1 for various cases of relative ordering between base values, pertinent positives and negatives and the feature values.

KDE Binning: The binning technique used to determine the grid points is the most essential part of the algorithm. We actually would like to place the grid points such that it creates equally spaced intervals such that every interval has equal probability. Consider the j-th feature x_j. Suppose, L_j and U_j are lower and upper bounds for this feature. We estimate the marginal density of this feature using a KDE estimate by using an appropriate kernel with a bandwidth on the points taken from this feature and obtain a Cumulative density function $P_j(x)$. Suppose $N + 1$ is the number of grid points we desire. Now, using root finding techniques, we actually find $\frac{k}{N}$-th quantile for $k \in \{1, 2, 3..N - 1\}$. This grid generation procedure is given in Algorithm 2.

Learning Algorithm: We assume that a base learner (BL) is given to us like a Decision Tree learner or a Logistic Regression based learner. Then algorithm 3 learns a transparent model based on the boolean rules/features extracted using GBFL.

4 Experiments

We now empirically validate our method. We first describe the setup, followed by a discussion of the experimental results. We provide quantitative results as well as present the most important boolean features picked by the base learners based on our construction.

Algorithm 1 Global Boolean Feature Learning (GBFL)

Input: Training set:D , Binning Matrix: G, Base Values vector: $\mathbf{b}$, Pertinent Positives: $\mathbf{p}(D)$, Pertinent Negatives: $\mathbf{n}(D)$, Skip Parameter: Δ.
Output: A set of Boolean clauses $\mathcal{F}$.
for Training sample $(\mathbf{x}, y) \in \mathcal{D}$ **do**
  $Q = 1$
  for j in $[1 : d]$ **do**
    if $p[x_j] > b_j$ **then**
      $g* \leftarrow \underset{g_{sj}: b_j < g_{sj} < p(x_j)}{\arg\min} |g_{sj} - p(x_j)|$.
      $Q \leftarrow Q * \mathbf{1}(z_j \geq g*)$
      $s* \leftarrow \underset{s: g_{sj} > x_j}{\arg\min} |g_{sj} - x_j|$.
      $Q \leftarrow Q * \mathbf{1}(z_j < g_{s*+\Delta})$
    end if
    if $p(x_j) < b_j$ **then**
      $g* \leftarrow \underset{g_{sj}: b_j > g_{sj} > p(x_j)}{\arg\min} |g_{sj} - p(x_j)|$.
      $Q \leftarrow Q * \mathbf{1}(z_j < g*)$
      $s* \leftarrow \underset{s: g_{sj} < x_j}{\arg\min} |g_{sj} - x_j|$.
      $Q \leftarrow Q * \mathbf{1}(z_j \geq g_{s*-\Delta})$
    end if
    if $x_j > b_j$ **then**
      $g* \leftarrow \underset{g_{sj}: n(x_j) > g_{sj} > x_j}{\arg\min} |g_{sj} - n(x_j)|$.
      $Q \leftarrow Q * \mathbf{1}(z_j < g*)$
      $s* \leftarrow \underset{s: g_{sj} < x_j}{\arg\min} |g_{sj} - x_j|$.
      $Q \leftarrow Q * \mathbf{1}(z_j \geq g_{s*-\Delta})$
    end if
    if $x_j < b_j$ **then**
      $g* \leftarrow \underset{g_{sj}: n(x_j) < g_{sj} < x_j}{\arg\min} |g_{sj} - n(x_j)|$.
      $Q \leftarrow Q * \mathbf{1}(z_j > g*)$
      $s* \leftarrow \underset{s: g_{sj} \geq x_j}{\arg\min} |g_{sj} - x_j|$.
      $Q \leftarrow Q * \mathbf{1}(z_j < g_{s*+\Delta})$
    end if
  end for
  $\mathcal{F} \leftarrow \mathcal{F} \cup Q$
end for

Algorithm 2 KDE based Grid Point Generation (GPG)

Input: Number of grid points: N, Dataset: D, Bounds on Features:$\{L_j, U_j\}_j$,Bandwidth for KDE estimator: B.
Output: Grid matrix $G \in \mathbb{R}^{N+1 \times d}$
for j in $[1:d]$ **do**
  Collect the set of points $\mathcal{P}_j$ from feature j from dataset D.
  Obtain the KDE estimate $p_j(x) = \frac{1}{B|\mathcal{P}_j|} \sum_{p \in \mathcal{P}_j} K(\frac{x-p}{B})$
  Obtain the CDF function for this $P_j(x)$.
  Set lower and upper bounds to the extreme grid points: $G_{j0} \leftarrow L_j$, $G_{jN} \leftarrow U_j$.
  for n in $[1 : N-1]$ **do**
    Use root finding to set $G_{jn} \leftarrow \{x : P_j(x) = \frac{n}{N},\ L_j \leq x \leq U_j\}$
  end for
end for

Algorithm 3 Model Generation using Local Explanations

Grid Matrix $G \leftarrow \mathsf{GPG}(N, D, \{L_j, U_j\}_j, B)$
Use dataset D, a pre-trained black-box model C, find pertinent positives $\mathbf{p}(D)$ and negatives $\mathbf{n}(D)$ for every training sample using (1) and (2).
Obtain the set of boolean clauses:

$$\mathcal{F} \leftarrow \mathsf{GBFL}(D, G, \mathbf{b}, \mathbf{p}(D), \mathbf{n}(D))$$

Form a binary dataset $D_{\mathcal{F}} \in \{0,1\}^{|D| \times |\mathcal{F}|}$ whose rows are obtained by evaluating a training point $\mathbf{x} \in D$ through all clauses $C \in \mathcal{F}$.
Fit a base learner to this new dataset: $\mathsf{BL}(D_{\mathcal{F}})$.

Table 1: Dataset characteristics, where N denotes dataset size and d is the dimensionality.

Dataset	N	d	# of Classes	Domain
Sky Survey	10000	17	3	Astronomy
Credit Card	30000	24	2	Finance
WDBC	569	31	2	Healthcare
Higgs Boson	250K	29	2	Physics
Waveform	5000	21	3	Signal Proc.

4.1 Setup

We experimented on five publicly available datasets from Kaggle and UCI repository namely; Sky Survey, Credit Card, WDBC, Higgs Boson and Waveform. The dataset characteristics are in Table 1. For the Sky Survey dataset [22], which has three classes we also did binary classification tasks by considering pairs of classes, as those were also deemed as interesting during the Kaggle competition that this data was part of. The Higgs Boson dataset has 250 thousand training points. We randomly subsampled 20 thousand points and trained the classifiers.

Random Forest with 100 trees (RF) was taken to be the black-box classifier and logistic regression (with L1 penalty) trained on the original features is the base learner for Sky Survey and Credit Card datasets. A four layered deep neural network (DNN) with fully connected layers ($100 \rightarrow 50 \rightarrow 10 \rightarrow$ softmax) was chosen as the black-box model for WDBC, Higgs Boson and Waveform datasets with a decision tree (height $<= 5$) being the base learner. This shows the wide applicability of our approach to different black-box models as well as base learners that may or may not be differentiable. Statistically significant results that measure performance based on paired t-test are reported computed over 10 randomizations with 75/25% train/test split. 10-fold cross-validation is used to find all parameters.

Table 2: Below we see the performance of the different methods. BB stands for black-box model. Best results based on paired-t test comparing base learner accuracy to GBFL are in bold. * indicates we match the black-box models accuracy.

Dataset	BB Model	BB Accuracy	Base Learner (BL)	BL Accuracy	GBFL Accuracy
Sky Survey	RF	0.99	Logistic	0.90	**0.94**
Sky Survey (Star vs Galaxy)	RF	0.99	Logistic	0.91	**0.93**
Sky Survey (Galaxy vs Quasar)	RF	0.99	Logistic	0.96	**0.99***
Sky Survey (Quasar vs Galaxy)	RF	0.99	Logistic	0.92	**0.94**
Credit Card	RF	0.70	Logistic	0.67	**0.69**
WDBC	DNN	0.95	Decision Tree	**0.91**	**0.91**
Higgs Boson	DNN	0.70	Decision Tree	0.63	**0.68**
Waveform	DNN	0.85	Decision Tree	**0.75**	0.70

4.2 Quantitative Evaluation and Implications

We see a few noticeable trends from Table 2. Firstly, our method actually improves the performance of the base learner in many cases. This means that the constructed (sparse) boolean features from the local contrastive explanations have

valuable information about the prediction task over and above what the dataset offers. This could have interesting implications from at least two perspectives: 1) building accurate interpretable/transparent models that are robust and 2) from a privacy perspective, where one may not want to reveal too much information about their model. From the first perspective, our approach provides an avenue to leverage black-box models and corresponding local explanations to build a transparent model which could be deployed in high-stakes decision making. The models are also likely to be robust as they are based on boolean features that are non-differentiable. From the second perspective, not only are we somewhat replicating the black-box models performance, but since our model is transparent and based on its local explanations we might be revealing intricate details regarding its functioning that even a human user may be able to understand and replicate. This may not be acceptable to the model owner.

Secondly, we see in general that bigger the gap in performance between the black-box model and the base learner trained on the original dataset more the relative improvement. This is not too surprising, as if the local explanations are fidel to the black-box model they should contain rich information that cannot be readily extracted from the original dataset, at least using simple base learners.

Thirdly, the good generalization shown by GBFL implies that the contrastive explanations themselves seem to capture relevant information and hence, GBFLs performance could be a testament to overall quality of the explanations in terms of characterizing the model. It could thus be seen as a (global) quantitative metric to evaluate such as explanations, since locally both PPs and PNs always lie/do not lie in the predicted class.

4.3 Qualitative Evaluation

We now show boolean features constructed by our method for some of the datasets using the contrastive explanations for the respective black-box models that the base learner deems as most important. We see that although the boolean features are composed of multiple input/original features, the resultant model is still transparent and reasonably easy to parse. Definitely more so than the original black-box model.

Sky Survey Dataset In Listing 1, we see the top 4 boolean features based on L1-Logistic which is the base learner. We observe that nine out of the 17 original features were selected by the contrastive explanations which are a union of the PP and PN features. We then constructed boolean features out of them using algorithm 1. The different boolean features thus have conditions i.e. upper and lower bounds on the same set of original features. We can also see that some of the original features such as *redshift*, *mjd* and *dec* have the same condition across multiple boolean features indicating that them along with their ranges are likely to be most important. On the other hand, *ra* has different ranges in most cases. The other features have in between redundancy relative to ranges. As can be seen each boolean feature is composed of multiple input features which may make the formulas complicated. However, they still are formulas which can be parsed and the final decision for an input could be traced by following the

conditions and noting which boolean features would be satisfied.

Listing 1: We present below the top 4 boolean rule based features used by Logistic regression (with L1 penalty) on the Sky Survey Dataset along with their feature importances. We chose to show 4 since the importances fell significantly beyond these.

```
GBFL Feature with coefficient value = 2.68

1.51 >= PCA1 >= 0.0 &
0.66 >= PCA2 >= 0.0 &
0.26 >= PCA3 >= 0.0 &
629.05 >= fiberid >= 419.36 &
0.80 >= redshift >= 0.0 &
233.35 >= ra >= 137.26 &
57481 >= mjd >= 42354.42 &
14.42 >= dec >= 0.0 &
1770.52 >= plate >= 0.0


GBFL Feature with coefficient value = 1.48

2.53 >= PCA1 >= 0.50 &
0.49 >= PCA2 >= 0.0 &
0.35 >= PCA3 >= 0.0 &
2213.15 >= plate >= 442.63 &
14.42 >= dec >= 0.0 &
0.80 redshift >= 0.0 &
262.10 >= fiberid >= 52.42 &
164.72 >= ra >= 109.81 &
57481 >= mjd >= 42354.42


GBFL Feature with coefficient value = 1.31

1.51 >= PCA1 >= 0.0 &
0.49 >= PCA2 >= 0.0 &
0.35 >= PCA3 >= 0.0 &
260.81 >= ra >= 233.35 &
0.80 >= redshift >= 0.0 &
524.21 >= fiberid >= 157.26 &
14.42 >= dec >= 0.0 &
1770.52 >= plate >= 0.0 &
57481 >= mjd >= 42354.42


GBFL Feature with coefficient value = 1.10

2.53 >= PCA1 >= 0.50 &
0.49 >= PCA2 >= 0.0 &
```

```
0.44 >= PCA3 >= 0.089 &
576.63 >= fiberid >= 366.94 &
260.81 >= ra >= 233.35 &
14.42 >= dec >= 0.0 &
0.80 >= redshift >= 0.0 &
1770.52 >= plate >= 0.0 &
57481.0 >= mjd >= 42354.42
```

WDBC In Listing 2, we see the top 3 boolean features based on a small decision tree as the base learner. Again, boolean features were constructed by parsing PPs and PNs for the training points. The number of original input features selected is more for this dataset than Sky Survey. Nonetheless, since they are still just boolean formulas the model is transparent and in this case too the decision for a data point can be traced by following the conditions. Here too features such as *n1_concavepts*, *n0_fractald*, *n1_concavity* and *n2_symmetry* are repeated in all the boolean features with the same range of values, which could be indicative these input features along with their ranges being important in the decision making. Other features either do not repeat or repeat but have different conditions/ranges.

Listing 2: We present below the top 3 boolean rule based features used by the decision tree on the WDBC dataset ranked by their importance. More boolean features are provided in the supplement.

```
GBFL most important feature

3575.86>=n2_area>=863.33 &
36.04>=n2_radius>=17.29 &
n1_fractald<=0.01 & n0_concavity<=0.28 &
n1_area<=363.87 & n1_compactness<=0.09 &
n1_concavepts<=0.03 & n0_symmetry>=0.13 &
n2_concavity<=0.83 & n2_fractald<=0.15 &
n0_area<=2108.08 & n0_smoothness<=0.14 &
n0_fractald>=0.04 & n0_concavepts<=0.16 &
n2_texture<=43.28 & n2_smoothness<=0.19 &
n0_perimeter<=188.5 & n2_symmetry>=0.15 &
n2_concavepts<=0.27 & n0_texture>=9.71 &
n0_compactness<=0.29 & n1_radius>=0.11 &
n1_texture>=0.36 & n1_perimeter>=0.75 &
n1_smoothness>=0 & n1_concavity>=0 &
n1_symmetry>=0 & n2_perimeter<=251.2 &
n0_radius>=10.5 & n2_compactness<=0.71


GBFL rank 2 feature

n0_concavity>=0.07 & n2_concavepts>=0.13 &
7.22<=n1_area<=274.71 &
```

0<=n1_fractald<=0.01 &
185.2<=n2_area<=2219.6 &
n0_perimeter<=140.26 &
0<=n1_compactness<=0.09 &
0<=n1_concavepts<=0.03 &
7.93<=n2_radius<=26.66 &
0.01<=n0_compactness<=0.29 &
0<=n0_concavity<=0.35 &
12.02<=n2_texture<=43.28 &
0.02<=n2_compactness<=0.88 &
0.05<=n2_fractald<=0.18 &
0.07<=n0_smoothness<=0.16 &
0.12<=n2_smoothness<=0.22 &
0.2<=n2_concavity<=1.25 &
0.09<=n2_concavepts<=0.27 &
21.06>=n0_radius>=6.98 &
n0_texture>=9.71 & n0_concavepts>=0.0 &
1322.25>=n0_area>=143.5 &
n0_fractald>=0.04 & 1.49>=n1_radius>=0.11 &
2.62>=n1_texture>=0.36 &
11.36>=n1_perimeter>=0.75 &
n1_smoothness>=0 & n1_concavity>=0 &
0.04>=n1_symmetry>=0 &
n2_perimeter>=50.41 &
n2_symmetry>=0.15 & n0_symmetry>=0.13

GBFL rank 3 feature

n0_concavity>=0.07 & n0_texture>=19.56 &
n1_area<=274.71 & n1_compactness<=0.06 &
n1_fractald<=0.01 & n2_compactness<=0.54 &
n2_fractald<=0.13 & n0_compactness<=0.23 &
n1_concavepts<=0.03 & n2_area<=2897.73 &
n2_concavity<=0.83 & n0_area<=2108.08 &
n0_smoothness<=0.14 & n0_concavity<=0.35 &
n0_concavepts<=0.16 & n1_smoothness<=0.01 &
n2_radius<=31.35 & n0_texture<=39.28 &
n0_perimeter<=188.5 & n2_texture<=49.54 &
n2_smoothness<=0.2226 & n0_symmetry>=0.106 &
n0_fractald>=0.04 & n1_radius>=0.11 &
n1_texture>=0.36 & n1_perimeter>=0.75 &
n1_concavity>=0 & n1_symmetry>=0 &
n2_perimeter>=50.41 & n2_symmetry>=0.15 &
n0_radius>=10.50 & n2_concavepts>=0.04

5 Discussion

As systems get more complicated (Neural Networks and Boosted Trees) replicating their performance using simple interpretable models might become increasingly challenging. Transparent models could be the answer here, where there is more leeway to build complicated models that can be traced for the decisions they make and hence are auditable. Auditability is extremely important in domains such as finance, where decisions need to be traceable and proxy models to explain black-boxes are not really acceptable [6]. Moreover, transparent boolean classifiers have an added advantage of efficiency where even large boolean formulas can potentially be made extremely scalable by implementing them in hardware.

Models built from local explanations showing good generalization is in some sense a true testament to the fidelity of the explanations and useful information they possess. Hence, accuracy of models built on our boolean features could provide a global view into the quality of these local explanations. On the flip side though, this could raise privacy concerns in terms of not only (mostly) replicating the performance of a proprietary black-box model, but also making its decisions transparent to a human who could gain unwanted insight into its functioning.

In the future, we would like to build other classifiers (viz. weighted rule sets) using our boolean features. Moreover, we would also like to study the theoretically reasons behind the good generalization provided by our method. We conjecture that this has connections to the stability results shown for stochastic gradient descent in deep learning settings, given that the local explanation method primarily relies on gradient descent.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[2] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504*, 2017.

[3] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

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

[5] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems 31*. 2018.

[6] Amit Dhurandhar, Tejaswini Pedapati, Avinash Balakrishnan, PinYu Chen, Karthikeyan Shanmugam, and Ruchir Puri. Model agnostic contrastive explanations for structured data. *arxiv*, 2019.

[7] Amit Dhurandhar, Karthikeyan Shanmugam, Ronny Luss, and Peder Olsen. Improving simple models with confidence profiles. *Advances of Neural Inf. Processing Systems (NeurIPS)*, 2018.

[8] Jeff Dean Geoffrey Hinton, Oriol Vinyals. Distilling the knowledge in a neural network. In *https://arxiv.org/abs/1503.02531*, 2015.

[9] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[10] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European Conference on Computer Vision*, 2016.

[11] Tsuyoshi Idé and Amit Dhurandhar. Supervised item response models for informative prediction. *Knowl. Inf. Syst.*, 51(1):235–257, April 2017.

[12] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[13] Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. *arXiv preprint arXiv:1905.07697*, 2019.

[14] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining*, 2016.

[15] Cynthia Rudin. Please stop explaining black box models for high stakes decisions. *NIPS Workshop on Critiquing and Correcting Trends in Machine Learning*, 2018.

[16] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[17] Su-In Lee Scott Lundberg. Unified framework for interpretable methods. In *In Advances of Neural Inf. Proc. Systems*, 2017.

[18] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *See https://arxiv.org/abs/1610.02391 v3*, 2016.

[19] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[20] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

[21] M. Sipser. *Introduction to the Theory of Computation 3rd.* Cengage Learning, 2013.

[22] SkyServer. Kaggle, 2018.

[23] Guolong Su, Dennis Wei, Kush Varshney, and Dmitry Malioutov. Interpretable two-level boolean rule learning for classification. In *https://arxiv.org/abs/1606.05798*, 2016.

[24] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR, 2017.

Supplement

Listing 3: We present below the top 5 boolean rule based features used by Logistic regression (with L1 penalty) on the Sky Survey Dataset along with their feature importances.

```
GBFL Feature with coefficient value = 2.68

1.51 >= PCA1 >= 0.0 &
0.66 >= PCA2 >= 0.0 &
0.26 >= PCA3 >= 0.0 &
629.05 >= fiberid >= 419.36 &
0.80 >= redshift >= 0.0 &
233.35 >= ra >= 137.26 &
57481 >= mjd >= 42354.42 &
14.42 >= dec >= 0.0 &
1770.52 >= plate >= 0.0


GBFL Feature with coefficient value = 1.48

2.53 >= PCA1 >= 0.50 &
0.49 >= PCA2 >= 0.0 &
0.35 >= PCA3 >= 0.0 &
2213.15 >= plate >= 442.63 &
14.42 >= dec >= 0.0 &
0.80 redshift >= 0.0 &
262.10 >= fiberid >= 52.42 &
164.72 >= ra >= 109.81 &
57481 >= mjd >= 42354.42


GBFL Feature with coefficient value = 1.31

1.51 >= PCA1 >= 0.0 &
0.49 >= PCA2 >= 0.0 &
0.35 >= PCA3 >= 0.0 &
260.81 >= ra >= 233.35 &
0.80 >= redshift >= 0.0 &
524.21 >= fiberid >= 157.26 &
14.42 >= dec >= 0.0 &
1770.52 >= plate >= 0.0 &
57481 >= mjd >= 42354.42


GBFL Feature with coefficient value = 1.10

2.53 >= PCA1 >= 0.50 &
0.49 >= PCA2 >= 0.0 &
```

```
0.44 >= PCA3 >= 0.089 &
576.63 >= fiberid >= 366.94 &
260.81 >= ra >= 233.35 &
14.42 >= dec >= 0.0 &
0.80 >= redshift >= 0.0 &
1770.52 >= plate >= 0.0 &
57481.0 >= mjd >= 42354.42

GBFL Feature with coefficient value = 0.54

2.53 >= PCA1 >= 0.50 &
0.49 >= PCA2 >= 0.0 &
0.35 >= PCA3 >= 0.0 &
10.82 >= dec >= 0.0 &
52.42 >= fiberid >= 0.0 &
178.44 >= ra >= 123.54 &
0.80 >= redshift >= 0.0 &
1770.52 >= plate >= 0.0 &
57481.0 >= mjd >= 42354.42
```

Listing 4: We present below the top 5 boolean rule based features used by the decision tree on the WDBC dataset ranked by their importance.

```
GBFL most important feature

3575.86>=n2_area>=863.33 &
36.04>=n2_radius>=17.29 &
n1_fractald<=0.01 & n0_concavity<=0.28 &
n1_area<=363.87 & n1_compactness<=0.09 &
n1_concavepts<=0.03 & n0_symmetry>=0.13 &
n2_concavity<=0.83 & n2_fractald<=0.15 &
n0_area<=2108.08 & n0_smoothness<=0.14 &
n0_fractald>=0.04 & n0_concavepts<=0.16 &
n2_texture<=43.28 & n2_smoothness<=0.19 &
n0_perimeter<=188.5 & n2_symmetry>=0.15 &
n2_concavepts<=0.27 & n0_texture>=9.71 &
n0_compactness<=0.29 & n1_radius>=0.11 &
n1_texture>=0.36 & n1_perimeter>=0.75 &
n1_smoothness>=0 & n1_concavity>=0 &
n1_symmetry>=0 & n2_perimeter<=251.2 &
n0_radius>=10.5 & n2_compactness<=0.71


GBFL rank 2 feature

n0_concavity>=0.07 & n2_concavepts>=0.13 &
7.22<=n1_area<=274.71 &
0<=n1_fractald<=0.01 &
```

185.2<=n2_area<=2219.6 &
n0_perimeter<=140.26 &
0<=n1_compactness<=0.09 &
0<=n1_concavepts<=0.03 &
7.93<=n2_radius<=26.66 &
0.01<=n0_compactness<=0.29 &
0<=n0_concavity<=0.35 &
12.02<=n2_texture<=43.28 &
0.02<=n2_compactness<=0.88 &
0.05<=n2_fractald<=0.18 &
0.07<=n0_smoothness<=0.16 &
0.12<=n2_smoothness<=0.22 &
0.2<=n2_concavity<=1.25 &
0.09<=n2_concavepts<=0.27 &
21.06>=n0_radius>=6.98 &
n0_texture>=9.71 & n0_concavepts>=0.0 &
1322.25>=n0_area>=143.5 &
n0_fractald>=0.04 & 1.49>=n1_radius>=0.11 &
2.62>=n1_texture>=0.36 &
11.36>=n1_perimeter>=0.75 &
n1_smoothness>=0 & n1_concavity>=0 &
0.04>=n1_symmetry>=0 &
n2_perimeter>=50.41 &
n2_symmetry>=0.15 & n0_symmetry>=0.13

GBFL rank 3 feature

n0_concavity>=0.07 & n0_texture>=19.56 &
n1_area<=274.71 & n1_compactness<=0.06 &
n1_fractald<=0.01 & n2_compactness<=0.54 &
n2_fractald<=0.13 & n0_compactness<=0.23 &
n1_concavepts<=0.03 & n2_area<=2897.73 &
n2_concavity<=0.83 & n0_area<=2108.08 &
n0_smoothness<=0.14 & n0_concavity<=0.35 &
n0_concavepts<=0.16 & n1_smoothness<=0.01 &
n2_radius<=31.35 & n0_texture<=39.28 &
n0_perimeter<=188.5 & n2_texture<=49.54 &
n2_smoothness<=0.2226 & n0_symmetry>=0.106 &
n0_fractald>=0.04 & n1_radius>=0.11 &
n1_texture>=0.36 & n1_perimeter>=0.75 &
n1_concavity>=0 & n1_symmetry>=0 &
n2_perimeter>=50.41 & n2_symmetry>=0.15 &
n0_radius>=10.50 & n2_concavepts>=0.04

GBFL rank 4 feature

nucleus2_area >= 863.33 & nucleus2_radius >= 17.29 &
nucleus1_compactness <= 0.06 & nucleus1_fractal_dim <= 0.01 &
nucleus2_fractal_dim <= 0.13 & nucleus1_area <= 363.87 &

nucleus1_concave_pts <= 0.03 & nucleus2_compactness <= 0.71 &
nucleus0_smoothness <= 0.14 & nucleus0_compactness <= 0.29 &
nucleus2_texture <= 43.28 & nucleus2_concavity <= 1.04 &
nucleus0_perimeter <= 188.5 & nucleus0_area <= 2501.0 &
nucleus0_concavity <= 0.42 & nucleus0_concave_pts <= 0.20 &
nucleus2_radius <= 36.04 & nucleus2_area <= 4254.0 &
nucleus2_smoothness <= 0.22 & nucleus0_texture >= 9.71 &
nucleus0_symmetry >= 0.10 & nucleus0_fractal_dim >= 0.04 &
nucleus1_radius >= 0.11 & nucleus1_texture >= 0.36 &
nucleus1_perimeter >= 0.75 & nucleus1_smoothness >= 0 &
nucleus1_concavity >= 0 & nucleus1_symmetry >= 0 &
nucleus2_symmetry >= 0.15 & nucleus0_radius >= 14.02 &
nucleus2_perimeter >= 117.33 & nucleus2_concave_pts >= 0.09

GBFL rank 5 feature

nucleus2_concave_pts >= 0.13 & 7.22 <= nucleus1_area <= 274.71 &
0 <= nucleus1_fractal_dim <= 0.01 & 185.2 <= nucleus2_area <= 2219.6 &
0.01 <= nucleus0_compactness <= 0.23 & 0 <= nucleus0_concavity <= 0.28 &
0 <= nucleus0_concave_pts <= 0.13 & 0 <= nucleus1_smoothness <= 0.01 &
0 <= nucleus1_compactness <= 0.09 & 0 <= nucleus1_concave_pts <= 0.03 &
7.93 <= nucleus2_radius <= 26.66 & 0.02 <= nucleus2_compactness <= 0.71 &
0 <= nucleus2_concavity <= 0.83 & 0.05 <= nucleus2_fractal_dim <= 0.15 &
43.79 <= nucleus0_perimeter <= 164.38 & 0.05 <= nucleus0_smoothness <= 0.14 &
0.07 <= nucleus2_smoothness <= 0.19 & 18.27 <= nucleus2_texture <= 49.54 &
0.04 <= nucleus2_concave_pts <= 0.27 & nucleus0_radius >= 6.98 &
nucleus0_texture >= 9.71 & nucleus0_area >= 143.5 &
nucleus0_symmetry >= 0.10 & nucleus0_fractal_dim >= 0.04 &
1.49 >= nucleus1_radius >= 0.11 & nucleus1_texture >= 0.36 &
nucleus1_perimeter >= 0.75 & nucleus1_concavity >= 0.0 &
0.04 >= nucleus1_symmetry >= 0 & nucleus2_perimeter >= 50.41 &
0.42 >= nucleus2_symmetry >= 0.15