Explanations for Multinomial Classifiers

Tips and Tricks for Practitioners

Pramit Choudhary[*]
Mountain View, CA
pramit.choudhary@h2o.ai

Navdeep Gill[*]
Mountain View, CA
navdeep.gill@h2o.ai

Patrick Hall[†]
Washington, DC
patrick.hall@h2o.ai

Abstract

1 Introduction

This short discussion bookends popular and practical texts on machine learning explanations by Choudhary, Gill, Hall et al. by specifically addressing the common and somewhat vexing problem of explaining the behavior and predictions of supervised multinomial classifiers [8], [7], [3] [17]. This paper is a continuation of ongoing research and our understanding of various techniques that are useful to understand a complex "black box" model's decision policies.

2 Notation

To facilitate technical descriptions of explanatory techniques, notation for input and output spaces, datasets, and models is defined.

2.1 Spaces

- Input features come from the set $\mathcal{X}$ contained in a *P*-dimensional input space, $\mathcal{X} \subset \mathbb{R}^P$.
- Known labels corresponding to instances of $\mathcal{X}$ come from the set $\mathcal{Y}$ contained in a *C*-dimensional input space, $\mathcal{Y} \subset \mathbb{R}^C$.
- Learned output responses come from the set $\hat{\mathcal{Y}}$. For classification models the set $\hat{\mathcal{Y}}$ typically contains a column vector for each unique class in $\mathcal{Y}$. In this text, the space $\hat{\mathcal{Y}}$ is said to be contained in a C'-dimensional output space, $\hat{\mathcal{Y}} \subset \mathbb{R}^{C'}$.

2.2 Datasets

- The input dataset $\mathbf{X}$ is composed of observed instances of the set $\mathcal{X}$ with a corresponding dataset of labels $\mathbf{Y}$, observed instances of the set $\mathcal{Y}$.
- Each i-th observation of $\mathbf{X}$ is denoted as $\mathbf{x}^{(i)} = [x_0^{(i)}, x_1^{(i)}, \ldots, x_{P-1}^{(i)}]$, with corresponding i-th labels in $\mathbf{Y}, \mathbf{y}^{(i)} = [y_0^{(i)}, y_1^{(i)}, \ldots, y_{C-1}^{(i)}]$, and corresponding predictions in $\hat{\mathbf{Y}}, \hat{\mathbf{y}}^{(i)} = [\hat{y}_0^{(i)}, \hat{y}_1^{(i)}, \ldots, \hat{y}_{C'-1}^{(i)}]$.
- $\mathbf{X}$ and $\mathbf{Y}$ consist of N tuples of observations: $[(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(N-1)}, \mathbf{y}^{(N-1)})]$.

[*]H2O.ai
[†]H2O.ai and George Washington University

Preprint. Work in progress.

- Each j-th input column vector of $\mathbf{X}$ is denoted as $X_j = [x_j^{(0)}, x_j^{(1)}, \ldots, x_j^{(N-1)}]^T$.

2.3 Models

- A type of machine learning model g, selected from a hypothesis set $\mathcal{H}$, is trained to represent an unknown signal-generating function f observed as $\mathbf{X}$ with labels $\mathbf{Y}$ using a training algorithm $\mathcal{A}$:
- g generates learned output responses on the input dataset $g(\mathbf{X}) = \hat{\mathbf{Y}}$, and on the general input space $g(\mathcal{X}) = \hat{\mathcal{Y}}$.
- The model to be explained is denoted as g.

3 Global Analysis

3.1 Data

Subsequent sections will use simulated data to empirically demonstrates the desired relationships and interaction between input feature space $\mathbf{X}$, the output space $f(\mathbf{X}) = \mathbf{Y}$. Simulated data is generated using a pre-defined signal-generating function using four input features with interactions and with eight noisy features with ground truth containing three unique categories(class 1, class 2, class 3) as defined below,

$$f = \text{num}_1 * \text{num}_4 + |\text{num}_8| * \text{num}_9^2 + e \tag{1}$$

A g_{GBM} is trained as the base estimator to learn the patterns represented using multinomial logloss as the optimization function and softmax is used to generate the final predictions. $\mathbf{X}, \mathbf{f}(\mathbf{X}) \xrightarrow{\mathcal{A}} g_{\text{GBM}}$ such that $g_{\text{GBM}} \approx f$.

Table 1: Summarizing evaluation during model training on the simulated dataset

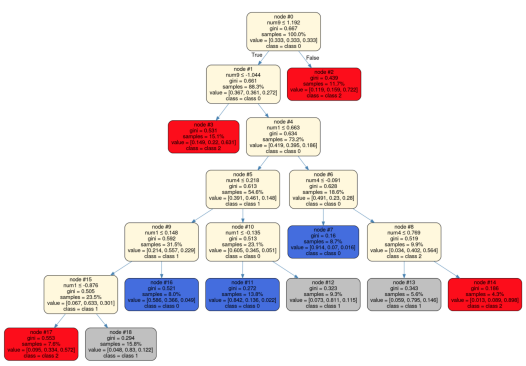
Dataset /Number of rows	F1-score	log-loss
Training set(20946)	0.79	0.56
Test set(9054)	0.70	0.71

** The performance metric reference in the 1 is for completeness. The goal of the paper is not to highlight the building the best possible model.

3.2 Interpretation via Decision Tree Surrogate

Given a learned function g and a set of learned output responses $g(\mathbf{X}) = \hat{\mathbf{Y}}$, a surrogate decision tree(SDT) h_{tree} can be constructed to extract comprehensible and faithful concepts such that $h_{\text{tree}}(\mathbf{X}) \approx g_{\text{GBM}}(\mathbf{X}) \approx f(\mathbf{X})$:

$$\mathbf{X}, g(\mathbf{X}) \xrightarrow{\mathcal{A}} h_{\text{tree}} \tag{2}$$



(a)

Ground Truth as provided: class-0.0

Predicted label using original GBM: class-1

Predicted label using using TreeSurrogate: class-1

```
As num9[0.7690225676118387] <= 1.192431926727295 then,
  As num9[0.7690225676118387] > -1.0435385704040527 then,
    As num1[-0.20470765948471295] <= 0.6633080840110779 then,
      As num4[-0.55573030434749] <= 0.2180337905883789 then,
        As num1[-0.20470765948471295] <= 0.14815638959407806 then,
          As num1[-0.20470765948471295] > -0.8763774037361145 then,
              Predicted Label: 1
```

(b)

Figure 1: **a:** Illustrates the surrogate decision tree built using the original estimator g_{GBM}. Only the leaf nodes are colored, clearly highlighting class membership. The non-leaf nodes highlights the important features(namely num1, num4, num9) which matches the results computed using the Shapely feature importance. **b:** Visualizes the decision stumps as a convenient text representation. The decision stumps highlight the rules learned by the SDT for a single input row. The nested "As-then" conditions contain the globally important features(num9, num1, num4). The ground truth for referenced input row is *"class-0"* but both the original model g_{GBM} and explanation model h_{tree} predicted *"class-1"*. Since, the explanation model has learned the original model's policies concisely, the explanations produced are a faithful approximate representation of the g_{GBM}

In the above mentioned learning task, the target labels are derived from the previously learned GBM response function g_{GBM}. [4] [1] provide detained information for training a tree surrogate h_{tree}.

3.2.1 Recommendations

- In order to prevent SDT from creating overly complex un-interpretable "if-then-else" rules that do not generalize well, pruning(pre-pruning with cross validation or post-pruning by optimizing on the multi-class evaluation metic - e.g. f1-score, categorical cross entropy) can be applied to build stable trees.

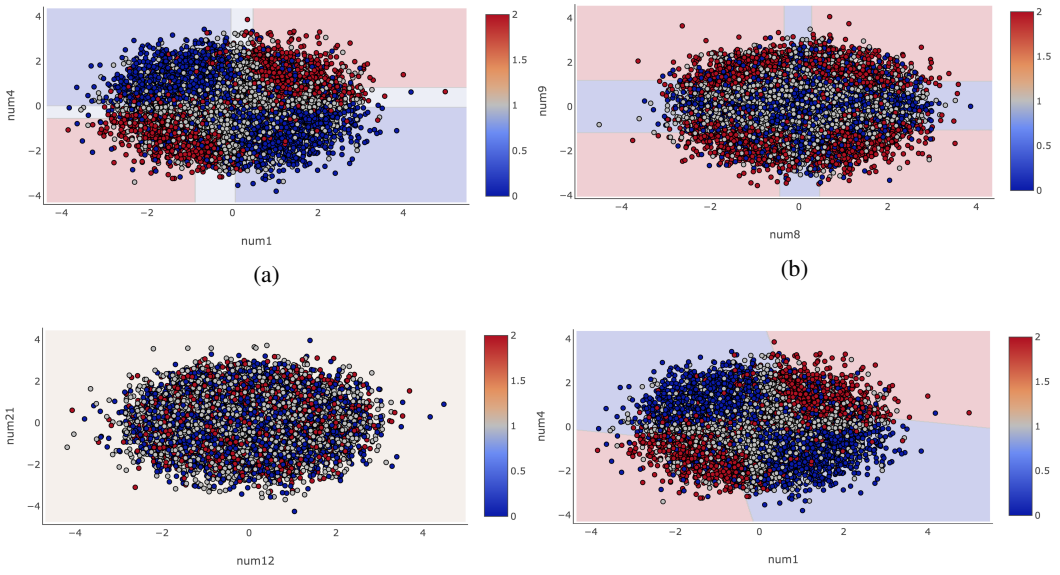
- SDT can elegantly handle high cardinality targets maintaining high level of fidelity and faithfulness to the original estimator function g). The faithfulness of the decisions generated by SDT depends on how precisely the tree surrogates captures the decisions learned by the original estimator.

- Remember to handle class imbalance before fitting a SDT using an oversampling technique - e.g. Synthetic Minority Over-sampling Technique(SMOTE) [2], Adaptive synthetic sampling approach for imbalanced learning(ADASYN)[9]

3.3 Interpretation via Decision Boundary Plots

Decision Boundaries is a hypersurface which provides informative interpretation to understand the characteristics of multiclass predictive model by visualizing the distances of the data elements to a model's learned decision plane. The boundary helps in providing a direct reference to the complexity of a classifier and the dataset [15] [13]. Figure 2 (a) and (d) illustrates different decision boundaries constructed using a GBM(Gradient Boosted Machine) and a MLP(Mulit-layer Perceptron classifier). Such representation provides global interpretation using the two or three dimensional feature vector X,

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3}$$



(a) (b) (c) (d)

Figure 2: The color of the data points correspond to the color of the supplied ground truth labels("class1", "class 2", "class 3"). While the background color of the decision boundaries indicated class membership(i.e. separation and direction) as learned by the GBM(**Figures a, b, c**). **Figure d:** Visualizes the decision boundary learned by a multi-layered MLP classifier, highlighting a different shape of the decision boundary as compared to GBM model(tree based models learn rectangular boundaries.)

3.3.1 Recommendations

- Decision Boundaries are widely used, easily understandable 2D or 3D scatter plots [13]
- Is able to show data and class membership in original feature space which is helpful in understanding feature interaction.
- Is model agnostic, and would work for any form of simple or complex("black box") model enabling effective model comparison.
- Provides the ability to instantly identify critical and cost changing data points, data elements closer to the decision boundary
- How do we handle high dimensional feature space $\mathcal{X} \subset \mathbb{R}^P$?
  - A matrix of 2D/3D decision boundary plots showing all combinations of the feature dimension. Top-K features could selected using the Shapely Feature Importance
  - Feature Selection/Extraction techniques could be applied to reduce the high-dimensional space - e.g. RFE, NMF, PCA, CCA, LDA

3.4 Interpretation via Shapley Global Feature Importance

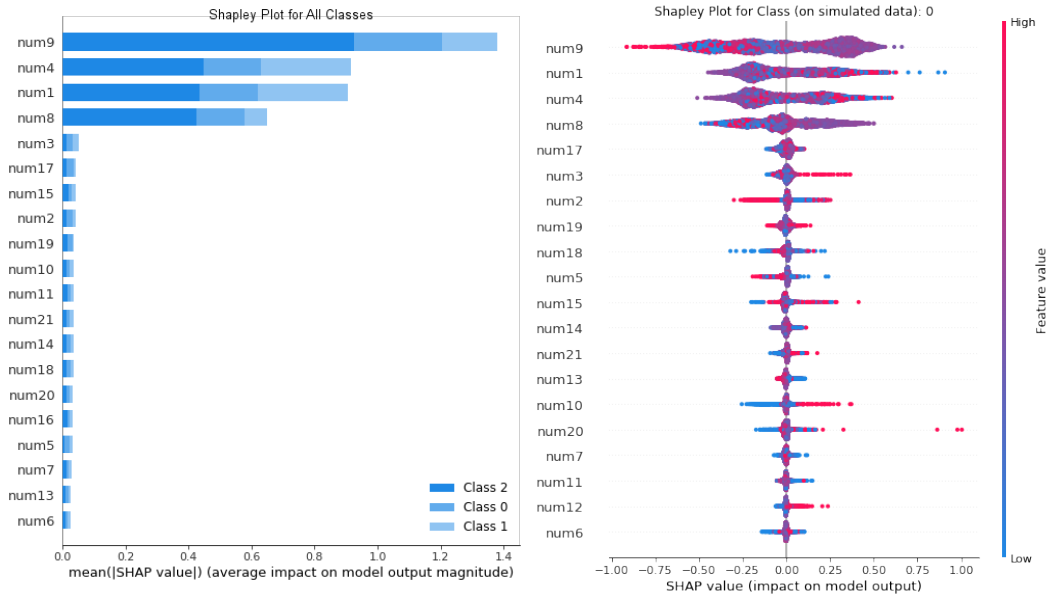
Shapley explanations, including tree shap and even certain implementations of LIME, are a class of additive, consistent local feature contribution measures with long-standing theoretical support [12]. Shapley explanations are the only possible locally accurate and consistent feature contribution values, meaning that Shapley explanation values for input features always sum to $g(\mathbf{x})$ and that Shapley explanation values can never decrease for some x_j when g is changed such that x_j truly makes a stronger contribution to $g(\mathbf{x})$ [12].

$$g(\mathbf{x}) = \phi_0 + \sum_{j=0}^{j=\mathcal{P}-1} \phi_j \mathbf{z}_j \tag{4}$$

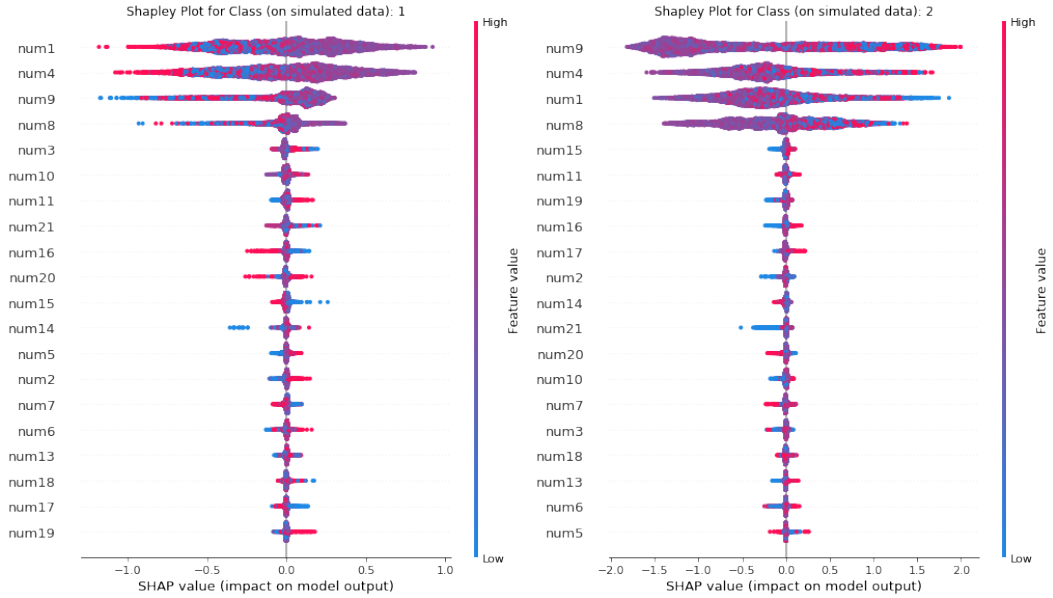
$$\phi_j = \sum_{S \subseteq \mathcal{P} \setminus \{j\}} \frac{|S|!(\mathcal{P} - |S| - 1)!}{\mathcal{P}!} [g_x(S \cup \{j\}) - g_x(S)] \tag{5}$$

Shapley values can be estimated in different ways. Tree shap is a specific implementation of Shapley explanations. It does not rely on surrogate models. Both tree shap and a related technique known as *treeinterpreter* rely instead on traversing internal tree structures to estimate the impact of each x_j for some $g(\mathbf{x})$ of interest [11], [16].

Simulated data is used to illustrate the utility of tree shap. Shapley explanations are estimated on $g_{\text{GBM}}(\mathbf{X})$ for a simulated test set $\mathbf{X}$ with known signal-generating function f. Results are presented in Figure **??**. Firstly, the Shapley explanations are shown globally across all class outcomes in a stacked bar chart broken down by absolute global Shapley values per class outcome. This is a good way to see a overall picture of Shapley explanations for multinomial classifiers. Secondly, the Shapley explanations are broken down per class outcome in subsequent charts. All feature contributions for $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 are seen as most important across all class outcomes both in the global stacked bar chart and per class outcome, which is expected based on Equation 1. However, they are not seen in the same order. For example, class 0 and class 2 share the same ordering of $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 but class 1 does not (num_1 and num_9 are swapped). This information can be used to investigate why the Shapley explanations differ between different class outcomes at a global level.



(a) (b)



(c) (d)

Figure 3

3.4.1 Recommendations

- Tree shap is ideal for estimating high-fidelity, consistent, and complete explanations of decision tree and decision tree ensemble models, perhaps even in regulated applications to generate regulator-mandated reason codes (also known as turn-down codes or adverse action codes).
- Because tree shap explanations are offsets from a global intercept, each ϕ_j can be interpreted as the difference in $g(\mathbf{x})$ and the average of $g(\mathbf{X})$ associated with some input feature x_j [14].
- What to do if very high cardinality in response, $\mathbf{Y}$:

– Examine top-K most frequent classes

– Examine top-K most accurate and inaccurate classes

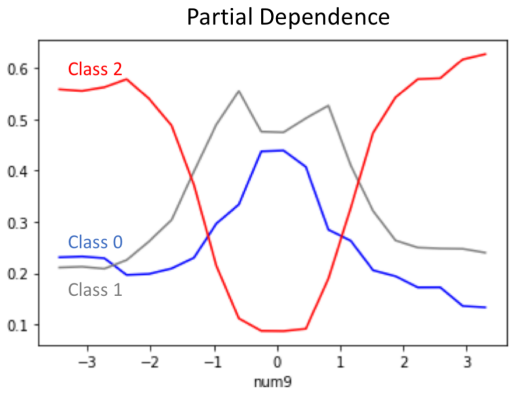
– Examine classes with highest variance in sum(absolute(shap))

3.5 Interpretation via Partial Dependence and ICE

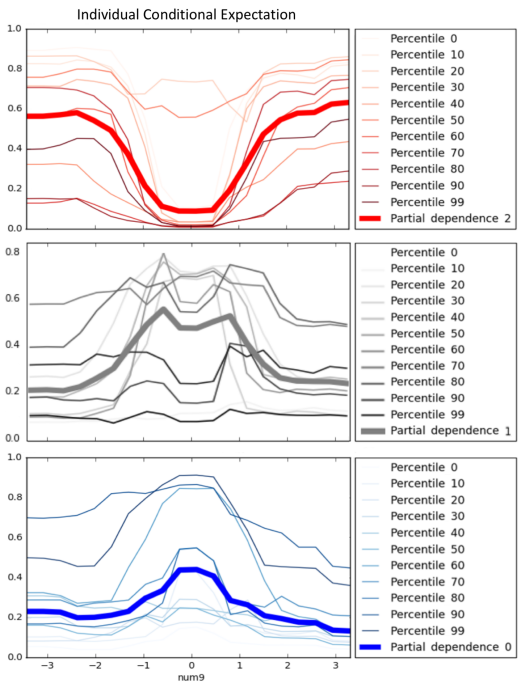
Partial dependence (PD) plots are a widely-used method for describing the average predictions of a complex model g across some partition of data $\mathbf{X}$ for some interesting input feature X_j [5]. Individual conditional expectation (ICE) plots are a newer method that describes the local behavior of g for a single instance $\mathbf{x} \in \mathcal{X}$. Partial dependence and ICE can be combined in the same plot to identify interactions modeled by g and to create a holistic portrait of the predictions of a complex model for some X_j [6].

Following Friedman et al. a single feature $X_j \in \mathbf{X}$ and its complement set $\mathbf{X}_{(-j)} \in \mathbf{X}$ (where $X_j \cup \mathbf{X}_{(-j)} = \mathbf{X}$) is considered. PD$(X_j, g)$ for a given feature X_j is estimated as the average output for a particular class outcome, C', of the learned function $g(\mathbf{X})$ when all the components of X_j are set to a constant $x \in \mathcal{X}$ and $\mathbf{X}_{(-j)}$ is left unchanged. ICE$(x_j, \mathbf{x}, g)$ for a given instance $\mathbf{x}$ and feature x_j is estimated as the output for a particular class outcome, C', for $g(\mathbf{x})$ when x_j is set to a constant $x \in \mathcal{X}$ and all other features $\mathbf{x} \in \mathbf{X}_{(-j)}$ are left untouched. Partial dependence and ICE curves are usually plotted over some set of constants $x \in \mathcal{X}$.

As in Section 3.2, simulated data is used to highlight desirable characteristics of partial dependence and ICE plots. In Figure **??** partial dependence and ICE at the minimum, maximum, and each decile of $g_{\text{GBM}}(\mathbf{X})$ are plotted per response outcome. The known quadratic behavior of num_9 is plainly visible, except for low/high value predictions across certain deciles per response outcome. When partial dependence and ICE curves diverge, this often points to an interaction that is being averaged out of the partial dependence. Given the form of Equation 1, there is a known interaction between num_9 and num_8. Combining the information from partial dependence and ICE plots with h_{tree} can help elucidate more detailed information about modeled interactions in g.



(a)



(b)

3.5.1 Recommendations

- Combining h_{tree} with partial dependence and ICE curves per class outcome is a convenient method for detecting, confirming, and understanding important interactions in g.
- What to do if very high cardinality in response, $\mathbf{Y}$:
  - Examine top-K most frequent classes
  - Examine top-K most accurate and inaccurate classes
  - Examine classes with highest variance in partial dependence
  - Examine classes with largest differences between partial dependence and ICE

4 Supplementary Materials

UCI credit card dataset [10].

`https://github.com/navdeep-G/interpretable-ml/tree/master/notebooks/credit/multinomial`

5 Conclusion

Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments go at the end of the paper. Do not include acknowledgments in the anonymized submission, only in the final paper.

References

[1] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting Blackbox Models via Model Extraction. *arXiv preprint arXiv:1705.08504*, 2017. URL: `https://arxiv.org/pdf/1705.08504.pdf`.

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[3] Pramit Choudhary. Interpreting Predictive Models with Skater: Unboxing Model Opacity. *O'Reilly Ideas*, 2018. URL: `https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity`.

[4] Mark W. Craven and Jude W. Shavlik. Extracting Tree-structured Representations of Trained Networks. *Advances in Neural Information Processing Systems*, 1996. URL: `http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks.pdf`.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. ***The Elements of Statistical Learning***. Springer, New York, 2001. URL: `https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12.pdf`.

[6] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics*, 24(1), 2015. URL: `https://arxiv.org/pdf/1309.6392.pdf`.

[7] Patrick Hall. On the Art and Science of Machine Learning Explanations. *arXiv preprint arXiv:1810.02909*, 2018. URL: `https://arxiv.org/pdf/1810.02909.pdf`.

[8] Patrick Hall, Wen Phan, and Sri Satish Ambati. Ideas on Interpreting Machine Learning. *O'Reilly Ideas*, 2017. URL: `https://www.oreilly.com/ideas/ideas-on-interpreting-machine-learning`.

[9] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008.

[10] M. Lichman. UCI Machine Learning Repository, 2013. URL: `http://archive.ics.uci.edu/ml`.

[11] Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. Consistent Individualized Feature Attribution for Tree Ensembles. *arXiv preprint arXiv:1802.03888*, 2018. URL: `https://arxiv.org/pdf/1706.06060.pdf`.

[12] Scott M. Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. URL: `http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf`.

[13] MA Migut, Marcel Worring, and Cor J Veenman. Visualizing multi-dimensional decision boundaries in 2d. *Data mining and knowledge discovery*, 29(1):273–295, 2015.

[14] Christoph Molnar. *Interpretable Machine Learning*. christophm.github.io/interpretable-ml-book, 2018. URL: `https://christophm.github.io/interpretable-ml-book/`.

[15] J Sunil Rao and William JE Potts. Visualizing bagged decision trees. In *KDD*, pages 243–246, 1997.

[16] Ando Saabas. Interpreting Random Forests, 2014. URL: `http://blog.datadive.net/interpreting-random-forests/`.

[17] Dipanjan Sarkar, Raghav Bali, and Tushar Sharma. Practical machine learning with python, 2018.