DRIL: Descriptive Rules by Interactive Learning

Category: Research

Figure 1: DRIL has four main parts: (A) the control panel, (B) the visualization panel, (C) the visualized rule list, and (D) the data table. The control panel handles data uploading and visualization options. The visualization panel is where the user selects data of interest for automatic characterization. The rule list includes features for adjusting thresholds and examining the relationships between the rules and the data. The data table is the detail view of raw data for reference.

Abstract

Analyzing data is increasingly a part of jobs across industry, science and government, but data stakeholders are not necessarily experts in analytics. The human-in-the-loop (HIL) approach includes semantic interaction tools, which leverage machine learning behind the scenes to assist users with their tasks without engaging them directly with algorithms. One widely applicable model for how humans understand data is descriptive rules, which can characterize important attributes and simultaneously their crucial values or ranges. In this paper, we introduce an approach to help with data understanding via interactively and automatically generated rules.Our approach makes discerning the behavior of groups of interesting data efficient and simple by bridging the gap between machine learning methods for rule learning and the user experience of sensemaking through visual exploration. We have evaluated our approach with machine learning experiments to confirm an existing rule learning algorithm performs well in this interactive context even with a small amount of user input, and created a prototype system, DRIL, to demonstrate its capability through a case study.

Keywords: human-in-the-loop, data characterization, rule learning

1 Introduction

Data analysis plays an increasingly important and broad role in industry, academia and government. People apply analytics and machine learning to understand patterns in their data to great effect. However, usually, the data stakeholders are not machine learning experts. This disconnect between skill sets is a key problem in unlocking the potential of data mining in broader industry. Training stakeholders in machine learning or machine learning experts in the domain is expensive and time-consuming. One solution, the human-in-the-loop (HIL) analytics approach, is to make the machine learning tools work for the domain experts behind the scenes, so the experts can make the best possible use of their knowledge and data.

One way that people aim to make sense of data is to look for the patterns "behind the data", i.e. connections between variables or groups of entities that are not obvious. Discovering natural groupings is a general functionality for understanding patients, customers, products, experimental outcomes, and more. Humans can describe insight about these groups by characterizing what separates those data from the rest. Using rules to specify these characteristics, e.g. "this high performing group of stores has low overhead and situates in a rural area", can be "models for human problem solving" [39]. Discovering these rules by hand can be impractical. In this work we provide a technique that leverages machine learning for automatic rule generation into an interactive mechanism for the user to intuitively control the process. In a feedback loop, the data stakeholder uses visualizations to spot potentially interesting groups of points. Without needing to explicitly direct any algorithms, they can get a fast characterization of these data. Through an interactive visualization of these descriptive rules, they can explore the relationship to the overall data and refine, rapidly developing their understanding of the data.

Our technique is explained in section 3, and our prototype implementation in section 4. The evaluation includes (1) machine learning experiments with simulated users to demonstrate how the underlying algorithms can be effective in our interactive context (section 5) and (2) case studies that demonstrate how this technique

can help to quickly reveal interesting patterns in data (section 6). In our Discussion (section 7), we explain the experiments used to choose a rule learning algorithm.

2 Related Work

This paper builds on work in interactive machine learning or human-in-the-loop (HIL) analytics for its core ideas of serving non-expert users in analytics tasks with behind-the-scenes algorithms. We also discuss maching learning algorithms for learning rules automatically.

2.1 Sensemaking with Interactive Machine Learning

Due to the broad desire to apply data mining in industry, there are many tools intended to make it easy to run machine learning algorithms. Commercial and free software includes applications with drag and drop interfaces for running algorithms or even designing big data pipelines [6, 14, 19, 22, 26].

In the visual analytics space, there has been work on assisting users of machine learning algorithms to tune parameters, especially semi-automatically, e.g. seeing live effects of changing dimension weights while running PCA [24], building regression models while incorporating domain expertise in feature selection [32], and choosing regression model parameters interactively based on their performance characteristics [13]. Due to its use of rules, RuleMatrix [31] is particularly relevant. It uses rules to help users understand behavior of other predictive models.

All of these tools still require the user to engage in some form with the machine learning. With a human-in-the-loop approach built around *semantic interaction*, however, we can broaden the audience further. The user performs implicit model steering and interacts with domain-appropriate visual tools while the machine learning is used behind the scenes to improve the workflow [16, 17]. A number of frameworks and surveys have covered HIL systems and the closely related concepts of interactive machine learning and human-centered machine learning [8, 18, 25, 34, 35]. The surveys provide many examples of techniques in this space, with applications like grouping people [3], learning distance measures [9], ranking [37], network alarm triage [4] and mental workload [1]. The technique we describe in this work has a similar structure, but contributes a new modality of semantic interaction where groups of points of interest can be quickly characterized to aid in rapid sensemaking.

2.2 Rule Learning

From the machine learning side, there are a wide variety of approaches for *rule-based classification*, in which IF-THEN rules define the prediction of a class. We first differentiate from *association rule mining*, which analyzes itemsets (sets of objects) for small common subsets. Rule-based classifiers are a better fit for our application and can be divided into *direct*, which learn rules directly from data (e.g. OneR [20], AQ [21], CN2 [11] and RIPPER [12]), and *indirect*, which learn rules from another model, like decision trees [33] or association rules [2] [36].

Another distinction in the types of rule learners is the difference between learning standalone rules and rule lists, wherein evaluation flows through the list until a classification decision is reached. We considered multiple algorithms and chose two to evaluate in detail with experiments for our interactive rule learning context, as discussed in section 7.

3 Interactive Rule Learning for Sensemaking

As described in , we aim to facilitate sensemaking for data stakeholders without analytics expertise. Descriptive rules are a "model for human problem solving" [39] since they are a convenient way to understand and describe patterns with simple statements broken down by relevant variables. Creating such rules by hand, scouring through variables for similarities across a group of points, is impractical. We use machine learning behind the scenes to help efficiently discover

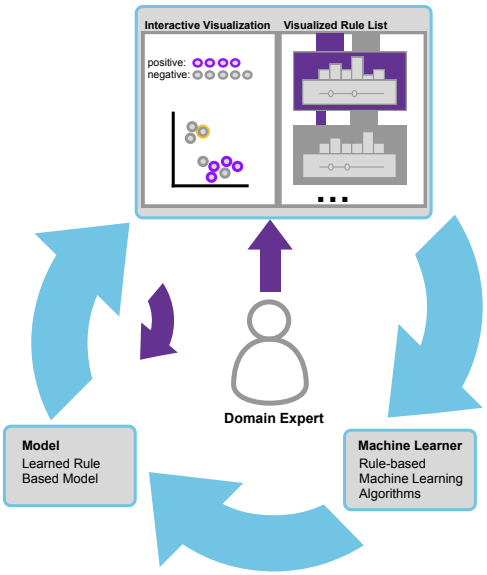

Figure 2: Diagram of our Human-in-the-Loop approach to sensemaking via rule learning. The domain expert interacts with a visual system in a sensemaking loop where they select points to see automatically generated rules that differentiate those points from the rest of the data. The rules are shown with interaction mechanisms to further explore their relationships with the data and iterate the process. See Section 3 for details.

descriptive rules. The user implicitly steers model development without needing to engage directly with machine learning.

In this section we explain how we adopt an existing learning algorithm for this purpose, including how we derive its inputs from user interaction and how we show its outputs back to the user. The diagram of Figure 2 shows the feedback loop of this HIL system, where the interactive visualizations help the user refine understanding, adjust data selection and refine the rules. This section explains our technique, but not the specifics of the prototype built to evaluate it, which are in Section 4.

3.1 Generating Rules for the User

Since this HIL process is guided by the user, the first step is that the user takes advantage of visualizations to explore data and develop questions or hypotheses to investigate. This could simply be in the form of noticing data points that do not conform to expectations. With some understanding of the data domain, the user may be acutely aware of what data points are expected to group together or appear atypical. There are many ways this manifests visually, but for example, there could be a group of points sitting together on the periphery of an apparent cluster in a data projection. However the points are noticed, the user's question becomes, "What is going on with this group?" An answer might specify not only which variables are relevant to explaining those data, but what value ranges are included versus excluded. A rule learner can be used to specifically answer this question, with conditions describing group members like "$profit\_ratio > .76$ AND $inventory\_value > 2,110,000$".

Our implicit model steering, HIL approach makes it easy to use this capability. Given simply a selection of points from the user, we begin the HIL loop shown in Figure 2 by transforming the user input to a machine learning input. Rule learning algorithms are generally supervised machine learning classifiers, meaning they take a dataset, $x_i \in X$, and a corresponding class label, $y_i \in C$, for each x_i, and learn a model that predicts which class an unseen data point x belongs to. In this case, we construct the class label from the user selection: for each data point x_i, we assign class $y_i = 1$ if x_i has been selected, otherwise $y_i = 0$. Because this label specifically

differentiates group membership, the model that will be learned by any supervised approach will be tuned to identifying how to separate the selected points from the rest.

When using a classifier to make predictions, it is important to balance the number of data items with each label. In this case, we expect imbalance because we do not want to require the user to label many points. However, because the models will be used only to guide sensemaking, not to make predictions, this should not be a concern. This tool can help early in the sensemaking process to generate insights and understanding, leading to further validation. We show in experiments (section 5) and a case study (section 6), that a small selection by the user can lead to interesting investigations.

3.2 Visualizing and Interacting with Rules

The rules that come straight from the machine learning are not ideal for presenting to a non-expert user. In order to facilitate closing the HIL loop of Figure 2, we must visualize the rules and provide interactions to help the user deepen understanding.

In the anatomy of the response from a rule learner, the top level is a list of *rules*, each of which corresponds to predicting that a point belongs with the user's selection, or that it does not. Each rule contains one or more *conditions*, which are the specific attribute ranges that contribute to the characterization. For example, as shown in Figure 3, IF $drat > 3.15$ THEN $class = 0$ is a rule, while $drat > 3.15$ is its only condition and the range is $(3.15, inf)$. This rule's class is 0, so points that satisfy the rule (all its conditions) are predicted not to be in the selected group. This corresponds to the rule model describing the group by excluding points that satisfy these conditions. Rules can have multiple conditions connected by the AND operator. The OR operator is not used, but is unnecessary, as there would instead be multiple rules, each with its own class output.

Another complication of the generated rules is apparent "inverse-duplicate" conditions, where a condition and its opposite are both provided. For example, IF $x < 5$ THEN *in_group* might be accompanied by IF $x >= 5$ THEN *out_group*. This redundancy is removed unless one of the conditions is in a rule with other conditions, as in Figure 3. In this case keeping both makes it possible (via mouseover) to see the effect of the variable in isolation as well as with its fellow conditions. When the rules are independent, e.g. with Quinlan [33], removing one does not affect the others. Even when the order does not matter, sorting them sensibly is important for user efficiency, so we prioritize positive rules (selected point) and those that cover a higher proportion of the data. Additionally, if a user makes a selection based on a scatterplot, two trivial rules may result: specifying the range in the x and y axes. These rules are uninformative, but rather than remove them post-hoc, the attributes corresponding to the visualization can be excluded from the model learning.

Finally, no machine learning will get perfect prediction always. HIL systems employ a feedback loop for iterative improvement (see Figure 2). Interactive visualizations to permit exploring the relationships between the rules and data are key. For example, one might examine the effect of rule thresholds by comparing them to the variable distributions. The user can then adjust the selection and get updated rules to show refined relationships.

4 The DRIL Prototype

The technique introduced in this work is explained in Section 3. In this section, we explain our prototype system used to validate the technique. The details include the specific visualizations and interactions provided, plus implementation details.

DRIL has four main components seen in Figure 1: (A) control panel, (b) data visualization panel, (C) visualized rule list, and (D) data table. Together, these provide an interface for loading and visualizing data, selecting points to provide input to the rule learner, and visualizing rules and their relationships to data.

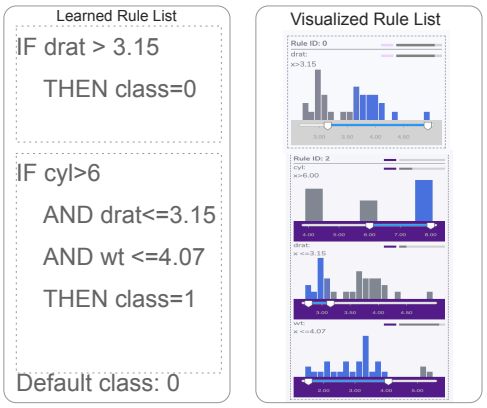

Figure 3: Output of a rule learner's output next to how it is visualized.

The control panel and visualization panel together are a straightforward interface for loading data, and visualizing it. The options include a projection of the data calculated with Multidimensional scaling (MDS) [7][1]. MDS is chosen over newer techniques like t-SNE [27] because t-SNE is meant to emphasize local structure and can create misleading spatializations sensitive to its multiple parameters [38]. A lasso selection is used for point selection. Once the analyst has chosen points of interest they click on the control panel to learn rules. DRIL will respond by learning a model to characterize the selected group. The mechanism for providing rules is described in Section 3.1, and we use Quinlan's method [33] as the rule learning in DRIL.

Certain transforms are needed to prepare the rules for visualization (see subsection 3.2). Each independent rule is bounded by a box with purple or gray trim to indicate a positive (point was selected) or negative rule, respectively. The mouse hover for these boxes highlights the corresponding data in the visualization panel, including marking incorrect in-group labels with a black stripe. Each rule box and condition within it is marked with two horizontal stacked bars to show the proportion of in-selection and non-selected data covered by the rule. Histograms or bar charts show how each condition's value boundaries relate to the distributions of the variables. The corresponding sliders and checkboxes allow the user to make adjustments. The effects of the adjustments can be seen when hovering.

5 Empirical Machine Learning Evaluation

We need to ensure that our learning method is capable of producing valuable rules for users with their limited interactions. This is not an evaluation of the machine learning algorithms themselves, which are established, but rather of how in our interactive environment they could support the task. In this section we explain how success is measured, and how we simulate different user interactions to collect performance data.

5.1 Metrics of Success

In a typical 2-class problem in machine learning, the framing of performance is through the tradeoff of precision vs. recall. These metrics model correctness at predicting a positive case and what proportion of actual positives were identified, respectively. In the case of our evaluation, we want to emphasize the ability to capture the set of points selected by the user (positives), so recall will be most helpful. However, over-prediction as positive will not be useful, so we also evaluate recall with the labels flipped, i.e. an "inverse

[1]Metric MDS computed with the python scikit-learn library [10] with min-max scaled data and Euclidean distance. Dummy variables are used for categoricals. Other parameters were defaults.

Figure 4: Case Study - Frankie quickly discovers some commonalities among employee absences. See section 6

.

recall" that measures the ability to predict all non-selected points correctly[2].

5.2 Simulation of Effectiveness for User

We assume that users need only select a small amount of data as input for the rule learning, and that they discover these as groups through visualization. Thus, two goals of the empirical machine learning evaluation are to evaluate how many selected points are sufficient, and to simulate user selection based on visual properties.

We conducted four types of simulations and measured the results with recall and inverse recall.The four methods are (1) *random selection*: user selects random points to select (a baseline), (2) *random closest selection*: user selects one point at random, then chooses the closest k points, (3) random closest selection in a cluster: user selects one point at random, and closest points within the same cluster[3] are chosen, and (4) *overall closest selection*: user chooses closest set of points in the entire dataset. Distances are Euclidean in the MDS space, to match what a person would see. For each simulation, we tested a selection size of 10, 20, 30, 40, and 50 points. Because of the random point selection, each experiment (corresponding to a tuple of simulation technique and number of points) was run 20 times and the results were averaged. We tested with six datasets common in machine learning literature (AbsenteeismAtWork [29], BreastCancer [28], Banknote [15], Heart [23], Ionosphere [5], and Sonar [5]), which have varying numbers of points and attributes.

The simulation results are shown in the supplementary materials, broken down by recall vs. inverse recall, then by simulation technique, with number of selection points in the x axis and a line representing each dataset. First, for *random closest positive points* and *random closest positive points in a cluster*, we see the expected pattern that increasing the number of points improves performance. We also note that performance is good at 10 labels (better than 80 percent) and consistently near-perfect after 20. For inverse recall, the denominator is much higher and the performance is near perfect for all levels of selected points. However, for *random selection* and *closest positive points*, the rule learning sometimes fails and the performance is less consistent. This is because randomly choosing points, which is included only as a baseline, can result in a selection that does not have enough signal to model. Similarly, the absolute closest set of points will perform poorly as the arbitrary number of points grows beyond the size of the tightest group.

6 Case Study

Frankie is a manager in a company and has collected data on absenteeism [30] and is looking for patterns that cause employees to miss work to inform project estimates. To begin, she wants a generic overview and selects an MDS projection so she does not have to pick individual variables (Figure 4). She chooses to map color to the amount of absentee time in hours, looking for the biggest cases. There are multiple apparent groups, which may correspond to types of absence behavior. Though the darkest-blue cases are spread around, she notices several in one of the groups and wants to investigate what explains these absences. DRIL makes this type of sensemaking trivial: she simply selects that set of points and clicks a button.

In response, a set of rules describing those selected data points appears to the right. DRIL has identified an interesting pattern: these absences are all during the Fall and Winter AND all people who identify as social drinkers. She wants to explore further, so she mouses over the negative rule that says non-social-drinker. The corresponding highlight shows that social drinker may itself be a helpful variable, as it separates the top half of the projection from the bottom. Additionally, it is clear that none of the people in the selected group are non-social-drinkers. This absenteeism may be due to people getting sick from social gatherings during flu season. Certainly, Frankie must do a confirmatory analysis before responsibly reporting such a hypothesis, but the aim of our technique is to facilitate the early sensemaking process where the user grows understanding of the data and develops hypotheses. Thanks to DRIL, Frankie has quickly been able to glean insight about what variables and specifically what ranges of those variables may be key factors in absenteeism. In DRIL, she can continue to explore other groups or visualize the data with these variables to see if there may be additional structure to explore.

7 Discussion - Choice of Rule Learning Algorithm

There are numerous options for using machine learning to discover rules, as outlined in section 2. We considered multiple approaches but planned to give the user a choice between an algorithm producing independent rules based on decision trees (Quinlan [33]) and one that produces decision lists based on direct optimization, (SBRL [40]).

There are important differences between these algorithms, including their ability to handle categorical data, their speed performance, and their sensitivity to poor or limited labels. We evaluated these techniques in our interactive context just as in section 5. Results from these experiments are provided in the supplementary materials. Quinlan performed so much better and more reliably *for this task*, especially considering the label imbalance common in this application, that we have not used SBRL in DRIL.

8 Conclusion

In conclusion, this paper has presented our approach to helping a data stakeholder without machine learning background efficiently grow understanding through automatic rule generation.The learning mechanism is behind the scenes and draws on existing ML algorithms. We evaluated our choice of algorithm as well as its performance in this interactive context with simulations. A proof-of-concept prototype, DRIL, shows how this technology can work with a practical implementation that allows us to show a case study of its effectiveness. Using rapid, interactive, automatic generation of rules in a human-in-the-loop system could be a broadly useful technique that can be incorporated into many applications.

[2]This "inverse recall" can also be called specificity, but that name does not describe its purpose straightforwardly.

[3]Hierarchical agglomerative clustering avoids the issue of choosing the number of clusters because we can choose it to get enough points.

REFERENCES

[1] D. Afergan, E. M. Peck, E. T. Solovey, A. Jenkins, S. W. Hincks, E. T. Brown, R. Chang, and R. J. Jacob. Dynamic difficulty using brain metrics of workload. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3797–3806, 2014.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216, 1993.

[3] S. Amershi, J. Fogarty, and D. Weld. Regroup: Interactive machine learning for on-demand group creation in social networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 21–30. ACM, 2012.

[4] S. Amershi, B. Lee, A. Kapoor, R. Mahajan, and B. Christian. Human-guided machine learning for fast and accurate network alarm triage. In *Proceedings of the International Joint Conference on Artifical Intelligence (IJCAI)*, pages 2564–2569, 2011.

[5] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[6] J. Barnes. Azure machine learning. *Microsoft Azure Essentials. 1st ed, Microsoft*, 2015.

[7] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer Verlag, 2005.

[8] E. T. Brown, A. Endert, and R. Chang. Human-machine-learner interaction: The best of both worlds. In *CHI Workshop on Human Centred Machine Learning (HCML)*, 2016.

[9] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-function: Learning distance functions interactively. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 83–92. IEEE, 2012.

[10] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[11] P. Clark and T. Niblett. The cn2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[12] W. W. Cohen. Fast effective rule induction. In *Machine learning proceedings 1995*, pages 115–123. Elsevier, 1995.

[13] S. Das, D. Cashman, R. Chang, and A. Endert. Beames: Interactive multimodel steering, selection, and inspection for regression tasks. *IEEE computer graphics and applications*, 39(5):20–32, 2019.

[14] J. Demšar, T. Curk, A. Erjavec, Črt Gorup, T. Hočevar, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, M. Štajdohar, L. Umek, L. Žagar, J. Žbontar, M. Žitnik, and B. Zupan. Orange: Data mining toolbox in python. *Journal of Machine Learning Research*, 14:2349–2353, 2013.

[15] D. Dua and C. Graff. UCI machine learning repository, 2017.

[16] A. Endert, P. Fiaux, and C. North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 473–482. ACM, 2012.

[17] A. Endert, C. Han, D. Maiti, L. House, S. Leman, and C. North. Observation-level interaction with statistical models for visual analytics. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 121–130. IEEE, 2011.

[18] A. Endert, W. Ribarsky, C. Turkay, B. Wong, I. Nabney, I. Díaz Blanco, and F. Rossi. The state of the art in integrating machine learning into visual analytics. *Computer Graphics Forum*, 3 2017.

[19] Frank, Eibe and Hall, Mark A and Witten, Ian H. *Data Mining: Practical Machine Learning Tools and Techniques*. The WEKA Workbench, 2016.

[20] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, 1993.

[21] J. Hong, I. Mozetic, and R. S. Michalski. Aq15: Incremental learning of attribute-based descriptions from examples: The method and user's guide. Technical report, 1986.

[22] International Business Machines (IBM). IBM SPSS decision trees 26, 2019. Retrieved 21 FEB 2020 from ftp://public.dhe.ibm.com/ software/analytics/ spss/documentation/ statistics/26.0/ en/client/ Manuals/ IBM_SPSS_Decision_Trees.pdf.

[23] A. Janosi, W. Steinbrunn, M. Pfisterer, and R. Detrano. Heart disease data set, 1988.

[24] D. Jeong, C. Ziemkiewicz, B. Fisher, W. Ribarsky, and R. Chang. iPCA: An interactive system for PCA-based visual analytics. *Computer Graphics Forum*, pages 767–774, 2009.

[25] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. *Visual analytics: Definition, process, and challenges*. Springer, 2008.

[26] F.-F. Li and J. Li. Cloud automl: Making ai accessible to every business. *Internet: https://www. blog. google/topics/google-cloud/cloud-automl-making-ai-accessible-everybusiness*, 2018.

[27] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[28] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990.

[29] A. Martiniano, R. Ferreira, R. Sassi, and C. Affonso. Application of a neuro fuzzy network in prediction of absenteeism at work. In *7th Iberian Conference on Information Systems and Technologies (CISTI 2012)*, pages 1–4. IEEE, 2012.

[30] A. Martiniano, R. P. Ferreira, and R. J. Sassi. Absenteeism at work data set, 2012. retrieved from https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work.

[31] Y. Ming, H. Qu, and E. Bertini. Rulematrix: Visualizing and understanding classifiers with rules. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):342–352, Jan 2019.

[32] T. Muhlbacher and H. Piringer. A partition-based framework for building and validating regression models. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):1962–1971, 2013.

[33] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[34] D. Sacha, M. Sedlmair, L. Zhang, J. Lee, D. Weiskopf, S. North, and D. A. Keim. Human-centered machine learning through interactive visualization: review and open challenges. In *In Proceedings of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2016.

[35] P. Y. Simard, S. Amershi, D. M. Chickering, A. E. Pelton, S. Ghorashi, C. Meek, G. Ramos, J. Suh, J. Verwey, M. Wang, et al. Machine teaching: A new paradigm for building machine learning systems. *arXiv preprint arXiv:1707.06742*, 2017.

[36] A. K. Tung. Rule-based classification., 2009.

[37] E. Wall, S. Das, R. Chawla, B. Kalidindi, E. Brown, and A. Endert. Podium: Ranking data using mixed-initiative visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 2017.

[38] M. Wattenberg, F. Viégas, and I. Johnson. How to use t-sne effectively. *Distill*, 2016.

[39] P. H. Winston. *"Artificial Intelligence, Third Edition*, chapter 8, page 172. Addison Wesley, 1984.

[40] H. Yang, C. D. Rudin, and M. Seltzer. Scalable bayesian rule lists. pages 3921—-3930, 2017.