Toward Efficient Automation of Interpretable Machine Learning

Boris Kovalerchuk, Nathan Neuhaus
Dept. of Computer Science
Central Washington University
USA
Boris.Kovalerchuk@cwu.edu, Nathan.Neuhaus@cwu.edu

***Abstract*—Developing more efficient automated methods for interpretable machine learning (ML) is an important and long-term machine-learning goal. Recent studies show that unintelligible "black" box models, such as Deep Learning Neural Networks, often outperform more interpretable "grey" or "white" box models such as Decision Trees, Bayesian networks, Logic Relational models and others. Being forced to choose between accuracy and interpretability, however, is a major obstacle in the wider adoption of ML in healthcare and other domains where decisions requires both facets. Due to human perceptual limitations in analyzing complex multidimensional relations in ML, complex ML must be "degraded" to the level of human understanding, thereby also degrading model accuracy. To address this challenge, this paper presents the Dominance Classifier and Predictor (DCP) algorithm, capable of automating the process of discovering human-understandable machine learning models that are simple and visualizable. The success of DCP is shown on the benchmark Wisconsin Breast Cancer dataset with the higher accuracy than the accuracy known for other interpretable methods on these data. Furthermore, the DCP algorithm shortens the accuracy gap between interpretable and non-interpretable models on these data. The DCP explanation includes both interpretable mathematical and visual forms. Such an approach opens a new opportunity for producing more accurate and domain-explainable ML models.**

Keywords—machine learning, explainability, interpretability, accuracy, classifier, visualization, visual model, dominant intervals.

I. MOTIVATION

The need to develop efficient automated methods for interpretable machine learning has long been recognized, however, in spite of significant progress, many fundamental challenges remain unresolved [7, 13]. Currently unintelligible "black" box models such as Deep Neural Networks often achieve higher accuracy than "grey" or "white" box models such as single Decision Trees, Bayesian Networks [18,13], propositional and First Order Logic rules [16,17] that are considered as being human understandable. Choosing between *accuracy* and *interpretability*, while acceptable in some applications, unfortunately has served as a major barrier in the wider adoption of machine learning models in healthcare; such as in breast cancer diagnosis, where the need to understand, validate, and trust decisions, requires both [12]. Therefore, the challenges are as follows: (1) how to produce *more explainable models*; (2) how to design the *explanation interface*; and (3) how to understand the psychological *requirements* for effective explanation [7, 12].

The ideal and unrealistic goal of "human-free" fully automated processes of interpretable Machine Learning is largely incompatible with the need to integrate human informal domain knowledge to ML models.

This paper proposes an algorithm, the **Dominance Classifier and Predictor (DCP),** which opens new opportunities for realistic automation via a combination of analytical computations along with visual knowledge discovery. The goal of automating this process is to produce *simple interpretable and visual ML classification models* which decrease the *gap* in accuracy between interpretable and complex non-interpretable black-box models. Below we demonstrate that this is achievable, even for tasks as important as cancer diagnostics, via the DCP algorithm.

Breast cancer, the leading cause of death of women between the ages of 40 - 55, is one of the most curable variants if caught early, with 5-year survival rates of stage-1, stage-2, stage-3, and stage-4, are 100%, 93%, 72%, and 22%, respectively [3]. State-of-the-art neural networks, now capable of achieving prediction accuracies greater than 98% [1, 2], seem attractive candidates for developing highly accurate and more interpretable diagnostic techniques, of which millions of lives depend on annually. Despite near perfect prediction accuracy shown in [1, 2], neural networks still succumb to the major limitations imposed on all black-box solutions; unintelligibility, hence have been deemed too risky to deploy in matters of life-and-death [5,6] where lack of interpretability leads to difficulties in discovering erroneous and dangerous predictions [4].

Interpretability arguably, then, is not just another dimension of assessing a given machine learning model's utility, but rather it is often the dimension to which all others are dependent. In highly regulated domains such as healthcare, banking, insurance, etc., where, for example, the General Data Protection Regulation of the European Union confers the *right to explanation* for algorithmic decisions [8], unintelligible

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

models, have little room at any level of accuracy being unable to operate within such narrow legal framework underpinned by a right to explanation. Conversely, methods able to provide intelligibility are increasingly important and in demand [7,20]. Such methods range from directly interpretable, to methods which combine interpretable and uninterpretable methods to get joint benefits, such as extracting interpretable rules from neural networks. However, these methods often are quite complicated.

The *complexity* of classification models produced by both interpretable and uninterpretable methods poses a significant challenge for human interpretation [12], due to *human perceptual limitations* with respect to analyzing complex relationships in high dimensional data. As a result, complex numerical ML models such as deep learning models and convolutional neural networks, chosen for their advantages in accuracy, must be actually be "*degraded*" to the level of human understanding to be put into practice, thus nullifying said advantages. Therefore, it was proposed in the literature [12], that by creating simplified explainable rules, humans can actually understand decisions made by such models. This is a major motivation for the proposed method to be *explainable and simple* for the human perception.

In this paper, we define a ML method as *interpretable* if the ML model it produces is human-readable, explainable and understandable in domain terms, i.e., is presented in terms of the *domain* from which the data are used to build said model. An example of such models are those which are produced via decision trees. In a more exact terms, interpretable models are those which are described as statements in propositional or in first order logic terms on predicates from the domain. A fully formal mathematical description can be found in [11].

The proposed DCP algorithm provides a human-friendly *visual explanation* for its models, in addition to the traditional *textual explanation* provided via natural language or mathematical forms. This opens further opportunities in filling the gap between informal tacit domain knowledge possessed by domain experts, and the ML model, fostering mutual insight.

Using the DCP algorithm on the Wisconsin Breast Cancer dataset, taken from the University of California at Irvine machine learning repository [19], we show a method for exceeding the level of accuracy achieved by other interpretable algorithms while maintaining interpretability and simplicity. Furthermore, our results reduce the gap between the accuracy of interpretable methods to that of higher accuracy of non-interpretable methods on the same dataset.

II. METHOD

The Dominance Classifier and Predictor (DCP) algorithm is presented below. The basis schema of the algorithm consists of the following steps: (1) constructing class dominance intervals, (2) combining intervals in the voting methods, (3) learning parameters of the algorithm, (4) visualizing the dominance structure and (5) explaining the prediction.

There are several different methods of constructing class dominance intervals. The most desirable method is finding "clean" intervals, which contain only cases of a single class. The next method consists of intervals with the following features: (i) clear dominance of cases of one class, (ii) relatively evenly distributed points of all classes in each interval, (iii) balanced training data for the number of cases in classes, (iv) relatively large the number of cases in each interval.

In reality, the situation is often far less desirable, with data often lacking many if these features. To compensate for this, "penalties" are introduced in the form of parameters and voting methods; the optimal values of which, learned during training. As a result, there are multiple versions of the DCP algorithm. This paper presents four of them.

A. Classifier Dominance Structure

The first step of the algorithm is producing the dominance classifier structure. The *classifier structure* is essentially a table containing intervals $\{V\}$ and the number of cases of each class on the training data that are within the respective interval. In the example in Table 1, the interval [0.1,0.3] on the predictor attribute X_i contains 10 cases of class 1 and 200 cases of class 2 while the interval [0.4, 0.5] on the same X_i contains 20 cases of class 1 and 10 cases of class 2.

TABLE 1. EXAMPLE: INTERVALS OF DISTRIBUTION OF CASES OF CLASSES IN AN ATTRIBUTE.

Interval in attribute X_i	**Number of cases of Class 1, h_{1i}**	**Number of cases of Class 2, h_{2i}**
[0.1,0.3]	10	200
[0.4, 0.5]	20	10
...	...	...

The steps of building the Classier Dominance Structure are below.

Step 1.1 consists of *sorting* each predictor attribute X_i by ascending value, then removing all duplicate values by grouping values. Figure 1 illustrates this step.

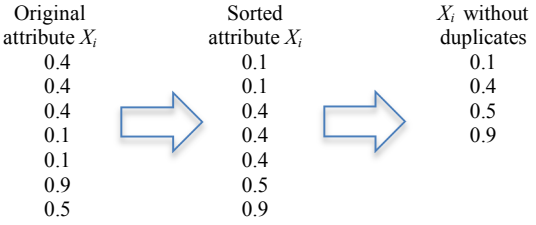


Fig. 1. Sorting and Removing Duplicate Values

Step 1.2 consists of *computing* the number of times each value appears by class for each predictor attribute. Figure 2 illustrates this step.

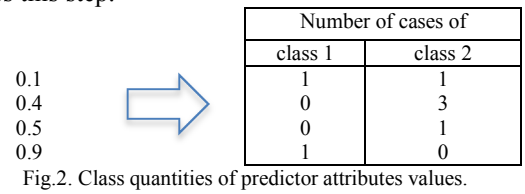


Fig.2. Class quantities of predictor attributes values.

Step 1.3 uses the information obtained in steps 1.1 and 1.2

for each predictor attribute to *calculate the dominant class* for each point, then groups all points into *intervals*, which are contiguous, dominated by the same class, and have a length greater than or equal to some *length threshold T* (see Figure 3).

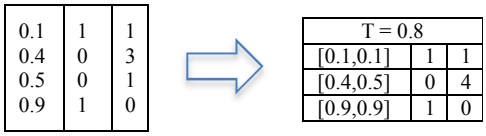


Fig. 3. DCP classifier constructed from class quantities and predictor attributes values

B. Classification Rules to Classify Cases

This part of the algorithm starts from **step 2.1** that is computing the *dominant class* in each attribute for a given case **x.** The *dominance level* is defined as the ratio of total number of points for class *j* to the total number of points of all classes. Given an arbitrary point, a search is conducted to ascertain whether an interval containing the point exists in attribute X_i.

If such an interval exists, the subsequent attributes are referenced to calculate the dominant class in said interval. If such an interval does not exist then no prediction is associated with the given attribute. For example, in a positive case, having value x_i=0.2, the algorithm finds interval [0.1,0.3], and then seeing that class 1 is represented by 10 values, and class 2 is represented by 100 values, class 2 being the greater of the two this interval would be used for class prediction for attribute X_i.

Step 2.2. After determining which class is dominant for the given $\mathbf{x}=(x_1,x_2,\ldots,x_n)$ for each x_i on the respective attribute X_i, the algorithm uses a voting method to *combine dominances*, based on individual attributes, to determine the total prediction. For example, for the input n-D point **x** with x_i=0.25 the use of Table 1 would activate interval [0.1,0.3], which in turn would predict class 2 over class 1 due to dominance on this interval with a much greater number of instances of class 2. For this **x**, class 2 is the *most dominant class* on X_i and interval [0.1, 0.3] is the *dominant interval* on X_i. Thus from such a table, a simple test of the value x_i can determine the corresponding interval V to be used for the prediction of the class of n-D point **x** based on the *most dominant class j* in this interval for the given attribute X_i with a simple one attribute **prediction rule R0**:

$$\text{If } \mathbf{x}=(x_1,x_2,\ldots,x_i,\ldots,x_n) \;\&\; x_i \in V \;\&\; h_j = \max(h_1(V),h_2(V),\ldots,h_k(V)) \text{ then } \mathbf{x} \in \text{class } j \quad (1)$$

where $h_1(V)$, $h_2(V)$,…,$h_k(V)$ are the numbers of cases of classes 1,2,…,*k*, respectively, that have values in interval V on attribute X_i.

The rule R_0 is based on a *single* attribute, while we have *n* attributes. We will use *notation* $j(\mathbf{x},i)$ to denote that class *j* is a dominant class for the n-D case **x** based on the attribute X_i. Next we will formulate rules that take into account *all n attributes* where each dominant class $d(\mathbf{x},i)$ gets some vote value. Below we consider *four voting methods* that differ in the number of votes assigned to each dominant class.

Each dominant interval gets one vote,

V_{ij}=1, if *j* is a dominant class on X_i, else V_{ij}=0

in a simple base method for *n* attributes, that we denote as **VM1**.

In VM1 the *total vote for class j* is a sum of votes for this class V_{ij} for all attributes X_i, it is the number of times when $d(\mathbf{x},i)=j$ by considering all attributes X_i,

$$V^{(j)} = \sum_{i=1}^{n} V_{ij} = |d(\mathbf{x},i)=j,\ i=1,n\}| \quad (2)$$

The classification decision is based on the comparison of total votes and finding their max value,

$$V^{(t)}(\mathbf{x}) = \max\left(V^{(1)}(\mathbf{x}), V^{(2)}(\mathbf{x}), \ldots, V^{(k)}(\mathbf{x})\right) \quad (3)$$

with the **classification rule, R1:**

$$\text{If } V^{(t)}(\mathbf{x})=\max\left(V^{(1)}(\mathbf{x}), V^{(2)}(\mathbf{x}), \ldots, V^{(k)}(\mathbf{x})\right) \text{ then } \mathbf{x} \text{ belongs to class } t \quad (4)$$

Thus, the voting method VM1 is resulted in formula (4). For each class *j* it checks that a given dimension's value x_i was classified to that class *j* (belongs to the interval that is dominated by class *j*), and if so, adds a single class-vote to class *j* votes, then find the max of these sums in formula (4). This method is problematic for many datasest, because it assumes that all attributes are equally predictive. In VM1 intervals containing a larger number of cases are equally predictive (or equally negligible) as intervals with fewer values, e.g., intervals with 4 points and 100 points when both dominated by the same class. Here we have an *underrepresented interval* for the class with only 4 points.

Other voting methods. Other voting methods that we denote as **VM2-VM4** assign votes V_{ij} and $V^{(1)}(\mathbf{x})$, $V^{(2)}(\mathbf{x})$,…., $V^{(k)}(\mathbf{x})$ in a more complex ways described below to mitigate weaknesses of VM1.

The **voting method VM2**, similar to the first one, however attempts to correct for the reliance on underrepresented intervals where predictability is assumed to be uniform. In VM2, a vote for the dominant class is the *number of class-members* in the interval. An interval containing 4 points from the single class, would add only 4 votes to the aggregate, and likewise an interval containing 100 points from the same class, would add 100 votes. While this method can resolve the *underrepresentation* issue, it may *over-represent* 100 cases by 100 votes assigned.

The **voting method VM3** is aimed to find the middle ground between high-accuracy intervals being underrepresented and single intervals being overrepresented. This method quantifies votes according to the *ratio of class-values to non-class-values*.

For example, if a given interval contains 95 benign-class values and 5 malignant-class values, instead of 95 class-votes being added to the aggregate, only 95/5=19 votes would be added.

The **voting method VM4** works in the same way that VM3 does, however, extends the method further in two ways: by *limiting the number of votes, via* a threshold, that any single attribute is allowed to cast to a maximum defined amount, and

via another threshold, by placing lower bounds on intervals lengths allowed to vote. For instance, if for attribute X_i the vote in method VM3 is 100 and the voting cap D for X_i is 10 then a new vote in VM4 for X_i will be 10 rather than 100. If a given interval length is 0.10, and the minimum allowed interval length is 0.20, then no vote will be counted for said interval. It seems that small intervals are generally less predictive than those covering a larger area which points to possible ways to go further via strengthening the voting methods by introducing more elaborate weighing mechanisms.

C. Learning, validation, visualization and explanation

The **step 3** of DCP algorithm is learning the parameters of the algorithm on training data to increase accuracy on these training data. The result of training is the tested on validation data with *10-fold cross validation*. The learning parameters that are described below include:

P1. Dominance Threshold Classifier parameter (DTC);
P2. Vote Limit Predictor parameter (VLP);
P3. Lower Length of the Interval (LLI).

The **DTC** threshold is the *minimum acceptable dominance ratio, D≥ DTC,* where D is the *dominance ratio* of a single class i over a given interval relative to other class j,

$$D=\mathrm{N}(i)/(\mathrm{N}(i)+\mathrm{N}(j)),$$

where N(i) and N(j) are the number of cases of classes i and j, respectively, on the given interval.

Only intervals that exceed DCT are used in the classifier. The base DCP algorithm learns DCT via brute-force, i.e. by testing different DCT values between 0 and 1 at some step, e.g., 0.1 for all intervals and recording the best one in accuracy of classification on training data.

While the search for a single DCP for all classes and attributes X_i decreases the search space, the further reduction of the search space is by searching only values in which a single class dominates at a ratio not less than or equal to 0.5 (50%), as any intervals with dominance ratio not greater than 50% is not dominant.

The **VLP** parameter is applied with the voting methods VM4. It is the *maximum number of votes* ("*voting cap"*) that any predictor attribute X_k may cast towards a given class. Let V_k (i) be the number of votes that X_k casts towards class i, then

$$\forall k\ V_k(i) \le \mathrm{VLP}.$$

The voting cap VLP is learned during training via brute-force, i.e. by testing which voting cap produces the most accurate prediction. The VLP is in the interval from 1 to the ratio N(s)/N(t), where N(s) is the number of cases of the *most dominant class* and N(t) is the number of cases of the second most dominant class. For instance, for N(1)=100, N(2)=20, N(3)=10 the VLP values are in [1,5] interval, because classes 1 and 2 are the first and second dominant with N(1)/N(2)=5.

The Lower Length of the Interval (**LLI**) is the *smallest length of the dominant intervals for which votes are counted*. LLI is learned during training via brute-force too, i.e. by testing different LLI values to find LLI with the most accurate prediction. The DCP algorithm normalizes all attributes X_i to [0,1] interval, therefore the LLI is in this interval. If the length L of a given interval is less than LLI, then the vote of this interval will not be counted. For instance, the vote on interval [0.05, 0.3] with L= 0.25 will not be counted when LLI =0.3.

The brute force learning of these parameters can be substituted by multiple optimization methods as well as by random search. To reduce the complexity of learning, instead of doing this for every possible combination of the above three parameters, a more time efficient *sequential search* is conducted where each parameter is optimized one after another.

For example, after finding the optimal dominance threshold, it is fixed and is used as a constant in the search for the optimal vote cap by brute-force. This process continues for LLI. A caveat to this approach, however, is that the order in which parameters are optimized could very well matter, and thus a sub-optimal result might be obtained, via such a heuristic.

The **steps 4 and 5** are visualization and explanation of the dominance structure and prediction using Parallel Coordinates and other General Line Coordinates (GLC) [9] that are illustrated in the next section for Parallel Coordinates.

III. CASE STUDY AND COMPARISON WITH PUBLISHED RESULTS

A. Case Study

Accuracy. The Wisconsin Breast Cancer dataset (WBCD) from the University of California at Irvine machine learning repository [19], was used in the following experiments. The dataset consists of 688 samples, and 9 features, each normalized to [0,1] interval.

The features are; clump thickness (F1), uniformity of cell size (F2), uniformity of cell shape (F3), marginal adhesion (F4), single epithelial cell size (F5), bare nucleoi (F6), bland chromatin (F7), normal nuclei (F8), and mitoses (F9). There are 2 classes, benign and malignant, with 444 and 239 samples, respectively. Table 2 shows average ten-fold cross validation accuracies with voting methods VM1-VM4.

TABLE 2. AVERAGE TEN-FOLD CROSS VALIDATION ACCURACIES WITH VM1-VM4

Voting method	Average accuracy in 10-fold cross validation
VM1	85%
VM2	90%
VM3	95%
VM4	97%

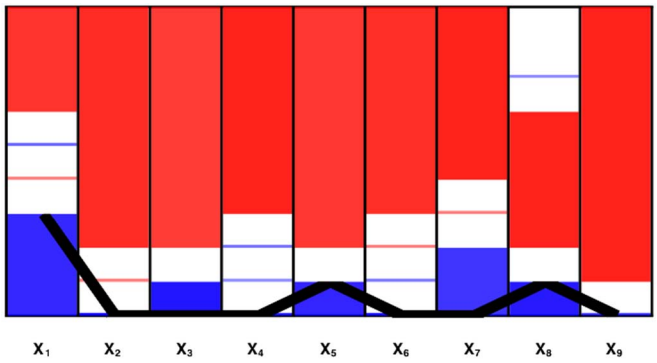
As this table shows the accuracy increases with moving to voting models that capture deeper relations in data by decreasing the *overrepresentation* of dominant intervals and attributes that contain too few point and *underrepresentation* of dominant intervals and attributes with a large number of points.

Table 3 shows in detail classification accuracies on the 10-fold cross validation tests with voting method VM4. The first

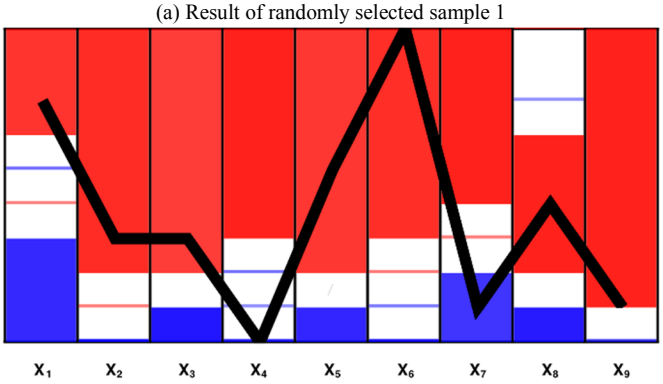
run achieved the highest classification accuracy of 98.5% with average accuracy of 97.01%.

TABLE 3. DETAILED TEN-FOLD CROSS VALIDATION RESULTS WITH VM4.

Run	Accuracy of 10-fold cross validation
1	0.985075
2	0.970149
3	0.985075
4	0.955224
5	0.985075
6	0.940298
7	0.940298
8	0.970149
9	0.985075
10	0.985075
Average	**0.9701**



(a) Result of randomly selected sample 1



(b) Result of randomly selected sample 2

Fig. 4. DCP visualization of dominant intervals with two randomly selected training cases (black lines) in Parallel Coordinates.

Visualization. Figure 4 shows the visualization of the structure of WBCD. Here each *vertical block* represents a single dimension X_i of the this cancer dataset with the values normalized to be in [0,1] interval. The blue color represents dominant intervals where majority attribute values belong to benign cases. The red color represents intervals where majority attribute values belong to malignant cases, and the white color indicates absence of values in the training set. This dominance structure is constructed in steps 1.1-1.3 of the DCP algorithm. It represents the data of the first 9 partitions of the 10-fold cross validation. Each column contains a noticeably large area absent of any values. Respectively the DCP algorithm will refuse to classify cases that have value in these intervals. The intensity of the colors allows to indicate the level of dominance and contribution of the interval to voting. The single red and blue polylines visualize single 9-D point (case) as it is visualized in Parallel Coordinates, The height of the nodes of these polylines are valued of respective attributes X_1,X_2, …, X_9. These examples are randomly drawn from the validation data. They are superimposed over the classification blocks to visualize exactly where the patient's data are malignant-like, and where these data are benign-like. It is visible that the case in Figure 4(a) is mostly in the benign areas and the case in Figure 4(b) is mostly in malignant area. These cases do indeed belong to the respective classes, and both predicted successfully.

Visual explanation. The visual explanation is based on Figure 4. Consider the black polyline (case) in Figure 4a, where it is visible that all its attributes X_1- X_9 are in the dominant intervals of the blue class. These dominant intervals have direct meaning in the breast cancer domain because they are in the original breast cancer attributes. This alone allows domain experts make professional judgments on them. Five of these intervals are wide ones that indicates stability of classification based on them to possible data measurement errors. In Figure 4b, the red case belongs to 7 red dominant intervals and only 2 blue dominant intervals, which also allows direct professional judgments on them by domain experts.

The **textual explanation** in the following form accompanies this visual explanation:

A [dimension-name] of [value] falls within in the know interval [x,y], which consists of *n* instances of class 1, and *m* instances of class 2.

The example is below for the first attribute normalized to [0,1]:

A normalized values of clump-thickness of 0.6 falls within the known interval [0.5, 0.7], which consists of 95 instances of class 1, and 5 instances of class 2.

Another type of explanation is one that we call **tabular pro-con explanation**. For a given *new case* ***c*** to be predicted, it shows *pro cases* (similar cases) and *con cases* (dissimilar cases) from both classes. This is a common explanation idea in k-nearest neighbors and case-based reasoning algorithms [13].

TABLE 4. TABULAR PRO-CON EXPLANATION BY DOMINANT INTERVAL CASES ON X_2.

Similar cases supporting class 1
0.33 0.11 0.00 0.00 0.11 0.00 0.11 0.00 0.00
0.22 0.11 0.00 0.00 0.00 0.00 0.11 0.00 0.00
0.55 0.11 0.00 0.00 0.00 0.00 0.66 0.00 0.00
0.11 0.11 0.11 0.00 0.00 0.00 0.66 0.00 0.00
0.44 0.11 0.00 0.00 0.11 0.00 0.22 0.00 0.00
0.22 0.11 0.00 0.00 0.11 0.11 0.22 0.00 0.00
0.33 0.11 0.00 0.00 0.11 0.11 0.22 0.00 0.00
0.00 0.11 0.11 0.00 0.11 0.00 0.11 0.00 0.00
Similar cases supporting class 2
0.44 0.11 0.22 0.33 0.11 0.66 0.22 0.55 0.00
0.77 0.11 0.33 0.00 0.44 0.00 0.44 0.33 0.33

In DCP the pro cases are cases with values of attributes in dominant intervals where case ***c*** belongs and the *con cases* are

from distant dominant intervals from the pro intervals. The demonstration of these cases relative to case ***c*** allows seeing the level of *consistency of prediction* of DCP algorithm for case ***c***. Table 4 shows the formal of this explanation with some pro cases on WBCD attribute X_2. Figure 4a visualizes one of the cases that support class 1. In this way, visual and tabular explanations complement each other.

B. Comparison with published results

Out of the 699 samples in WBC data, 16 were removed in preprocessing due to missing values, leaving 653 samples to be split between training and testing. This practice is consistent with the practice of most researchers using this dataset to allow the comparison of DCP to other interpretable ML methods by keeping other factors equal. Below we compare our results on 10-fold cross validation (CV) with results of other methods on 10-fold CV. Note that even in this case when all results are in 10 fold CV, we cannot make the exact comparison, because different authors make different splits of data to these 10 folds, in addition not all authors run their methods on all these folds. The comparison with other results listed in [20] that are not 10-fold cross validation would be incorrect and we avoid it. Interpretable C4.5, J4 and fuzzy *decision trees* produced results with accuracies of 94.74%, 94.36% and 95.27%, respectively, on 10–fold CV [14,15,18,20]. Different versions of less interpretable methods such as SVM and ANN produced more accurate results that range from 97.97% to 99.51% on the same 10–fold CV [20].

The average accuracy of 97.01% obtained by our DCP algorithm is higher than accuracies of interpretable methods reported in [14,15,18,20] that are in [94.36, 95.27] interval with average of 94.82. The average accuracy of 97.01% obtained by our DCP algorithm is below than accuracies of non-interpretable methods that are in [97.97, 99.52] interval with average of 98.74. The gap between these averages is 3.92. Our 97.01 differs from the average for interpretable models by 2.19 and by 1.73 from non-interpretable models. Thus, DCP algorithm decreases the gap more than two times.

While DCP algorithm did not reach the accuracy of the non-interpretable methods (SVM and ANN) its classification model is more interpretable and much simpler than models from those methods and with accuracy exceeded accuracy of interpretable decision trees on WBCD.

IV. Comparison of DCP with Decision Treee and Naïve Bayes Models

If out of *n* attributes, just one attribute exists with pure intervals (intervals that contain cases of a single class each) then this attribute alone is sufficient to classify training data with 100% accuracy. This is also a condition for non-overlapping of two hyper-rectangles. All other attributes and respective hyper-rectangles of cases of classes in n-D will play the role of *context*. An attempt to eliminate all of them from consideration can produce classification errors on new data that have values of these attributes outside of the training data.

An example is Iris data where a single attribute (petal length) allows full separation of class 1 (setosa) and class 2 (versicolor) [9,10]. An adversarial example with values of remaining three attributes (length and width of sepal, and petal width) ten times larger or smaller than in the training data still will be recognized as one of the iris types (setosa or versicolor) while it is not the case. It can be a very different flower or a completely artificial and non-existent object.

For the Decision Tree (DT) models, the presence of two pure intervals for two classes leads to a shortest possible decision tree with a single attribute and single split, i.e., a simple interpretable model, if $x_i < T$ then class 1 else class 2. Here T is any point between two intervals, e.g., T=1,5 for intervals [0,1] and [2,3].

This simple case allows showing the difference between DCP and DT. DT generalizes outside of actual intervals [0.1] and [2.3] to the area between them (1,2), where the first half of its point are classified to class 1 and other half to the class 2 when T=1.5.

The justification of T=1.5 instead of any other value from (1,2) is not trivial, if possible. Moreover, any T from (1,2) can produce overgeneralization with erroneous classification of cases between intervals, similarly to the Iris example above by classifying non-existent cases.

The same difference takes place in the situations without pure intervals, but with multiple dominant intervals. The next difference is the *voting* process that is not present in DT explicitly. In DT, all solutions are propositional statements such as

$$\text{If } x_1>5 \ \& \ x_2<8 \ \& \ x_3>4 \text{ then } \mathbf{x}=(x_1,x_2,x_3) \in \text{ class 1.}$$

that are potentially overgeneralizations as was shown above.

Let $\mathbf{x}=(x_1,x_2,x_3)$ and $x_1 \in [6,8]$, $x_2 \in [4,5]$ and $x_3 \in [5,9]$. In this situation, DCP provides richer and *more specific* interpretable information than DT on the same data via intervals such as

$\mathbf{x}=(x_1,x_2,x_3) \in$ class 1, because

$x_1=5 \in [3,6]$ on X_1 where class 1 dominates class 2 with ratio 10/1 & and this interval covers 75% of all class 1 training cases

$x_2=4.5 \in [4, 5]$ on X_2 where class 1 dominates class 2 with ratio 6/1 and this interval covers 90% of all class 1 training cases &

$x_3=7 \in [5,9]$ on X_3 where class 1 dominates class 2 with ratio 4/1 & this interval covers 65% of all class 1 training cases.

As we see, DCP explanation shows the length and representativeness of each dominance intervals used in the explanation. This allows the domain expert to judge how marginal the positon of a new case **x** is in the interval and how well the training data are represented in the dominant intervals.

The base voting in DCP is conceptually equivalent to constructing a disjunctive normal form (DNF). Let $P(x_i)$ be a property of x_i that is behind the vote for **x** with x_i for class 1. i.e., If $P(x_i)$=true then vote $V(\mathbf{x})$=1.

Consider a case of voting that 2 out 3 attributes votes for class 1 for $\mathbf{x}=(x_1,x_2,x_3)$. This voting is equivalent to the following DNF,

$$\text{If } [P(x_1) \,\&\, P(x_2) \text{ or } (P(x_1) \,\&\, P(x_3) \,\&\, (P(x_2) \,\&\, P(x_3)]$$

$$\text{then } \mathbf{x} \in \text{class 1}$$

Similarly, DNF can be constructed for the larger number of attributes, e.g., voting that 7 out 10 attributes votes for class 1. The long DNF is hard to discover and understand, but it is *easier to communicate* to domain experts as a voting statement.

Another difference is that DTs rules are sequential starting from the attribute at the root of the tree. DCP is invariant to the order of attributes. The sequential nature of DT has both advantages and disadvantages. The disadvantages led to development of forests of DTs to mitigate the dependence on the attribute selected to be at the root that may prohibit discovering some better rules. The advantage of DT over the DCP is in the same its sequential nature. DT makes next splits of the space that depend on the previous splits It allows discovering more refined rules than DCP discovers, but covering less number of cases possibly under-representative. Thus, a natural next step in developing enhanced DCP algorithm is adding mores splits and integrating with DT.

The *naïve Bayes* models first assume that for every n-D point **x,** the *probability* $P(C_i|\mathbf{x})$ that **x** belongs to class C_i can be *estimated* using available training data. The second assumption is that the attributes $x_1,x_2,..,x_n$ of **x** are *independent.* This allows getting $P(C_i|\mathbf{x})$ by combining these independent distributions. The third typical assumption is that these distributions are *Gaussian distributions,* with their means and standard deviations estimated from the training data.

The classification result, which is delivered to a user, from the naïve Bayes models is that **x** belongs to, say, class C_1, because the probability $P(C_1|\mathbf{x})$ for **x** to be in C_1 is greater than for any other class. It is *understandable and interpretable in the user's domain,* if it means that class C_1 is the *most frequent* class for all the cases, which are represented by the n-D point **x**. This statement would be correct, if all the probabilities $P(C_i|\mathbf{x})$ are estimated directly from the frequencies of n-D point **x** in the training data. Very rarely the training set is large enough for such an estimate. Thus, $P(C_i|\mathbf{x})$ are only the estimates made under assumptions, which are questionable or incorrect in the user's domain. In contrast, the DCP outputs presented above are better interpretable. It says that **x** belongs to class C_1, because the majority of the attributes are for this class, with specifics of the intervals on each attribute. Next, the user receives these attributes and intervals from DCP in the understandable visual form (see Figure 4) and can judge their dependence.

IV. CONCLUSION

This paper show that while we continue to be in the tradeoff between accuracy and interpretability, it is possible to decrease the gap from lower accuracy of interpretable methods to higher accuracy of non-interpretable methods and decreases the need in the tradeoff. DCP was able to cut out this gap more than two times on the benchmark Wisconsin breast cancer data, being much simpler than non-interpretable methods.

The following data factors define the application scope of DCP algorithm. It can success when dominant interval exists in data and dominant intervals are sufficient to construct a highly accurate model, i.e., subintervals and refined relations between intervals and subintervals are not required. The DCP algorithm allows testing presence of dominant intervals. Data that fail this test require other methods. The sufficiency of DCP depends on its ability to learn from discovered dominant intervals. The selection of a voting method and methods of discovering dominant intervals affects this ability to learn.

The future work is strengthening DCP by enhanced methods of discovering dominant intervals, complex relations between them and subintervals, and better learning strategies along with integration with Decision Trees. Our preliminary studies in this area show that this is a promising approach.

In general, the DCP opens further opportunities in filling the gap between informal tacit domain knowledge possessed by domain experts, and the ML model, fostering mutual insight.

V. REFERENCES

[1] Abdel-Zaher, A., Eldeib, A. Breast cancer classification using Deep Belief networks. Expert Systems with Applications Volume 42, Issue 10, 15 June 2015, 4611-4620.

[2] Bhardwaj A., Tiwari A. Breast cancer diagnosis using Genetically Optimized Neural Network model. Expert Systems with Applications Volume 42, Issue 10, 15 June 2015, 4611-4620.

[3] Breast Cancer Survival Rates. American Cancer Society. https://www.cancer.org/cancer/breast-cancer/understanding-a-breast-cancer-diagnosis/breast-cancer-survival-rates.html, Last accessed September 2018.

[4] Caruana, R., Lou, Y., Gehrke, J., Koch, P., et al, Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission. Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1721-1730, 2015.

[5] Cooper, G., Abraham, V., Aliferis, C., Aronis , J., et al, Predicting dire outcomes of patients with community acquired pneumonia. Journal of Biomedical Informatics, 38(5):347–366, 2005.

[6] Cooper, G., Aliferis, C., Aronis , J., Buchanan, B., Caruana, et al, An evaluation of machine-learning methods for predicting pneumonia mortality. Artificial Intelligence in Medicine, 9(2):107–138, 1997.

[7] DARPA, Explainable Artificial Intelligence, DARPA-BAA-16-53, 2016.

[8] General Data Protection Regulation, Article 22. Parliament and Council of the European Union, 2016.

[9] Kovalerchuk, B., Visual Knowledge Discovery and Machine Learning, Springer, 2018.

[10] Kovalerchuk, B. Grishin, V. Reversible Data Visualization to Support Machine Learning, In: S. Yamamoto and H. Mori (Eds.): HIMI 2018, LNCS 10904, Springer, 45–59, 2018.

[11] Kovalerchuk B., Vityaev E., Data Mining in Finance: Advances in Relational and Hybrid Methods 2000, Kluwer/Springer.

[12] Lakkaraju H., Bach S., Leskovec J. Interpretable decision sets: A joint framework for description and prediction. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining 2016 Aug 1, 1675-1684, ACM.

[13] Lipton Z. The Mythos of Model Interpretability, Communications of the ACM, 2108, 61,36-43.

[14] Pach, P., Abonyi, J. Association rule and decision tree based methods for fuzzy rule base generation. World Acad. Sci. Eng. Technol. 2006, 13, 45–50.

[15] Quinlan, J. Improved use of continuous attributes in C4.5. J. Artif. Intell. Res. 1996, 4, 77–90.

[16]Schüller P., Kazmi M. Best-Effort Inductive Logic Programming via Fine-grained Cost-based Hypothesis Generation. arXiv preprint arXiv:1707.02729. 2017 Jul.

[17]Speichert S., Belle V. Learning Probabilistic Logic Programs in Continuous Domains. arXiv preprint arXiv:1807.05527. 2018 Jul 15.
[18] Sumbaly, R., Vishnusri, N., Jeyalatha, S. Diagnosis of breast cancer using decision tree data mining technique. Int. J. Comput. Appl. 2014, 98, 16–24.
[19] Wisconsin breast cancer data, UCI. Machine Learning Repository. ftp.ics.uci.edu/pub/machine-learning-databases/breast-cancer-wisconsin.
[20] Yue, W., Wang, Z., Chen, H., Payne, A. and Liu, X., Machine Learning with Applications in Breast Cancer Diagnosis and Prognosis. Designs, 2(2), 2018, p.13.