Fall 2021

Full Interpretable Machine Learning Method with In-line Coordinates

Hoang Phan
Central Washington University, phanho@cwu.edu

FULL INTERPRETABLE MACHINE LEARNING METHOD

WITH IN-LINE COORDINATES

A Thesis

Presented to

The Graduate Faculty

Central Washington University

In Partial Fulfillment of the

Requirements for the Degree

Master of Science

Computational Science

---

by

Justin Phan

November 2021

CENTRAL WASHINGTON UNIVERSITY

Graduate Studies

We hereby approve the thesis of

Justin Phan

Candidate for the degree of Master of Science

APPROVED FOR THE GRADUATE FACULTY

______________ ____________________________

Dr. Boris Kovalerchuk

______________ ____________________________

Dr. Razvan Andonie

______________ ____________________________

Dr. Szilard Vajda

______________ ____________________________

Dean of Graduate Studies

ABSTRACT

FULL INTERPRETABLE MACHINE LEARNING METHOD

WITH IN-LINE COORDINATES

by

Justin Phan

November 2021

This thesis explores a new approach for machine learning classification task in 2-dimensional space (2-D ML) with In-line Coordinates. This is a full machine learning approach that does not require to deal with n-dimensional data in n-dimensional space. In-line coordinates method allows discovering n-D patterns in 2-D space without loss of n-D information using graph representation of n-D data in 2-D. Specifically, this thesis shows that it can be done with In-line Based Coordinates in different modifications, which are defined, including static and dynamic ones. Some classification and regression algorithms based on these In-line Coordinates were explored. Two successful cases studies based on benchmark datasets (Wisconsin Breast Cancer dataset and Page Block Classification dataset) demonstrated the feasibility of the approach. This approach helps to consolidate further a whole new area of full 2-D *machine learning with a respective methodology*. In-line coordinates method has advantages to actively include the end-users into the discovering of models and their justification. Another advantage is providing interpretable ML models.

Keywords— interpretable machine learning, classification, regression, visual knowledge discovery.

ACKNOWLEDGMENTS

I would like to express my deepest thanks of gratitude to Dr. Boris Kovalerchuk for supporting me with this project. I am sincerely grateful for his patient and willingness to answer all my questions that I struggled with despite his busy schedule. Because of this thesis work, I know that I will have a very strong tool for my future. I would also like to thank Dr. Razvan Andonie and Dr. Szilard Vajda for teaching me new things about Machine Learning Algorithms. A Multithreading lesson which I learned from the Operating System class with Dr. Szilard Vajda proves to be very useful in Machine Learning field. Finally, I would like to thank my classmates and family for supporting and motivating me throughout the last two years.

Sincerely

Justin Phan

TABLE OF CONTENTS

LIST OF TABLES

LIST OF TABLES (CONTINUED)

LIST OF TABLES (CONTINUED)

Table Page

LIST OF TABLES (CONTINUED)

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Interpretable Machine Learning (ML) is a major focus in Machine Learning domain these days [1, 2]. The approaches range from explaining black box models to building explainable models from scratch. One of the attractive options is building machine learning models using visual means. However, it is challenging, because data in machine learning are multidimensional, which we cannot represent graphically in 2-D coordinates. So, tools are needed which will allow to do this efficiently. Traditional methods which convert n-D data to two dimensions are lossy, not preserving all multidimensional information . In contrast representation of n-D data using General Line Coordinates (GLC) is lossless [3]. This visual representation opened the opportunity to do *full multidimensional machine learning in two dimensions without loss of information*. The advantage of this approach is two-fold. In simple situations, it can discover the pattern visually just by observing these data visualized in GLC. In more complex situations, which are common in ML, it can discover patterns in 2-D representations using new 2-D ML methods. It is growing into a whole new field of machine learning. This thesis is in this realm with the focus on a specific type of General Line Coordinates that is the In-line Coordinates [3]. This thesis is going to explain In-line Coordinates in chapter II.

In the series of prior work at CWU [4, 2, 5, 6, 7, 8] the feasibility of full 2-D ML with different types of General Line Coordinates (Shifted Paired Coordinates, Elliptic Paired Coordinates, CPC-R and GLC-L) were demonstrated. The works in this realm can traced to [9] for Parallel Coordinates and [10] for 2-D modeling of non-image data. Often 2-D studies in ML cover only simple 2-dimensional examples to illustrate ML algorithms visually. Next, visual analytics studies have been very active in exploring Parallel Coordinates for tasks related to clustering [11], but much fewer for supervised learning, which is the focus of our study. The works in this area include [12, 13, 14].

It was suggested in [2] to consolidate all such studies within a general concept, which can be called a **full 2D ML methodology**. Traditionally 2D studies in machine learning were considered as just auxiliary exploratory data/model visualization with loss of n-D information mostly afterwards or before the actual machine learning. It was assumed that in 2-D it is losing n-D information, and it needs full n-dimensional analysis in n-D space to construct ML models. The full 2-D ML methodology shows that it's not necessary. This methodology goes beyond visual knowledge discovery, which is advocated in [3]. It expands the studies from visual discovery by a human supported by ML methods, to a full scope of machine learning methods, for discovering full patterns analytically in 2-D.

Section II defines main concepts of In-line coordinates. Section III presents Box Classification (BC) algorithm, Linear Classification and Regression algorithm. Section IV covers the case study with a benchmark Wisconsin Breast Cancer (WBC) data from UCI ML repository [15] that demonstrates the feasibility of the approach. WBC data has 699 cases with 16 cases missing at least one attribute value. At this moment, cases with missing attributes are removed manually to test the efficiency of BC algorithm. A proper case consist of nine attributes. There are 444 benign cases and 239 malignant cases in WBC dataset. Section V presents results of the Page Block Classification (PBC) using BC algorithm and compares its results with other algorithms. Section VI presents the conclusions.

CHAPTER II

DEFINITION: IN-LINE BASED COORDINATES SYSTEM

To find the best possible visual representation of n-D data, this thesis attempts several mappings of coordinates to visual representations using different order of coordinates. The General Line Coordinates defined in [3] allow drawing n coordinates axes in 2-D in a variety of ways: curved, parallel, unparalleled, collocated, disconnected, etc. GLCs include **In-line Coordinates** (ILC) shown in Fig. 1, which are similar to Parallel Coordinates, except that the axes $x_1, x_2, \ldots x_n$ are horizontal, not vertical. All coordinates are collocated on the same line and might overlap. A sequence of directed curves or polylines satisfies the requirement of lossless representation of n-D point in 2-D. The curves/polylines of different heights and shapes can show additional information [3] such as the distance between adjacent attributes values, $|x_i - x_{i+1}|$ like it is done in Fig. 1 below. In-line Coordinates require the same number of nodes and links as Parallel Coordinates, which makes the scope of applicability of these methods similar.

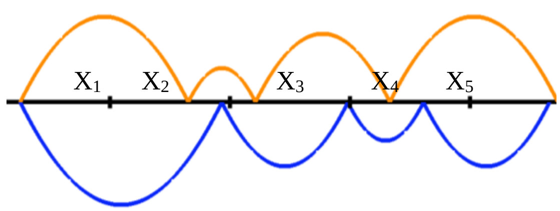


Fig. 1. Two 5-D points of two classes in In-line Coordinates.

The links between the nodes are directed edges, but arrowheads can be omitted when the direction follows the order of coordinates. To observe better the difference between n-D points of different classes, it can draw n-D points of one class above the coordinate line and n-D points of another class below (see Fig. 1). Tree location modes of ILC is considered:

(L1) *Sequential* ILC with coordinates located one after another (Fig. 1).

(L2) *Collocated* ILC with coordinates drawn at the same location with full overlap (Fig. 2a).

(L3) *Generic* ILC where some coordinates can be *sequential*, *collocated*, *overlapping*, or *disjoined* (Fig. 2b).

(L4) *Dynamic ILC* with coordinated located *dynamically* as it is explained later.

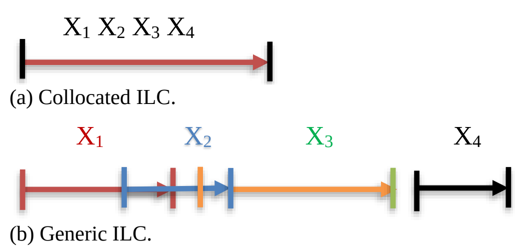
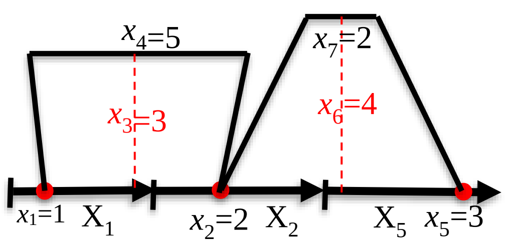


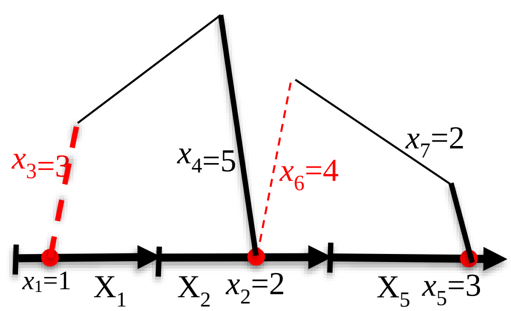
Fig. 2. Options to locate coordinates in In-line Coordinates.

In L3, a given n-D point **c** can be collapsed to a single 2-D point on ILC by selecting a specific ILC overlap [3]. It is a useful visual property when n-D point **c** is a center point of the class and other n-D points of this class are concentrated next to it. Reordering coordinates X_1-X_n is another option, to make the patterns of interest more visible.

There are several options to **construct links** that connect points x_i on coordinates X_i by assigning meaning to their characteristics such as its width and height to convey additional information. See an example in Fig. 3a for a 7-D point with values of x_3 and x_4 encoded as the **height** and the **width** of the line that connects (x_1,x_2), and values of x_6 and x_7 the height and width of the line that connects (x_2,x_5). Here, only three coordinates x_1, x_2 and x_5 are directly encoded in the base line of ILC making it shorter. Fig. 3b and Fig. 4 show other options. Fig. 3b uses the lengths of **sides** of the line that connects points x_1 and x_2 to encode values of x_3 and x_4, instead of using its width and height. Similarly, lengths of sides of the line that connects points x_2 and x_5 encode values of x_6 and x_7. The main goal of In-line Coordinates is supporting discovering n-D pattern and rules with highest possible values of precision and recall. Fig. 3 shows alternative designs of In-line Coordinates.



(a) x_3,x_4,x_6 and x_7 encoded by the height and width of the link lines that connect (x_1,x_2) and (x_2,x_5).

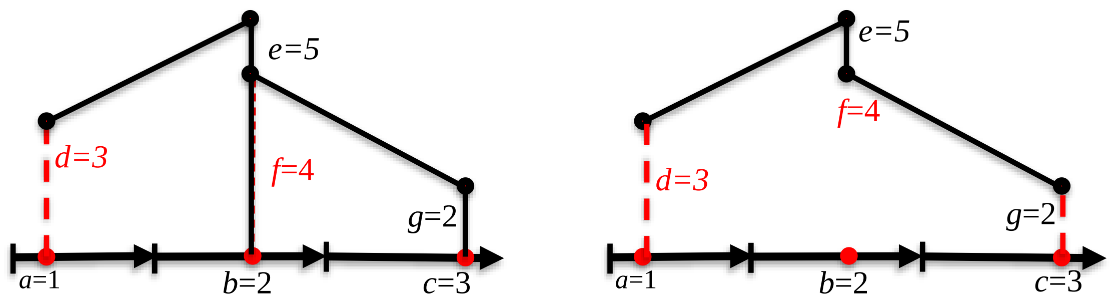


(b) x_3,x_4,x_6 and x_7 encoded by of length of sides of the link lines that connect (x_1,x_2) and (x_2,x_5).

Fig. 3. 7-D point **x** = (x_1,x_2,x_3,x_4,x_5,x_6,x_7) = (1,2,3,5,3,4,2) in two ILBCs.

These visual representations are not a pure ILC representation with a single baseline anymore but rather ILC based on it, therefore it is called **In-Line Based Coordinates** (ILBC) [2] which is presented below.

Fig. 4a simplifies Fig. 3b, by making sides **vertical,** and Fig. 4b simplifies this figure further by removing vertical lines, which go down to the baseline and keeping only solid lines.



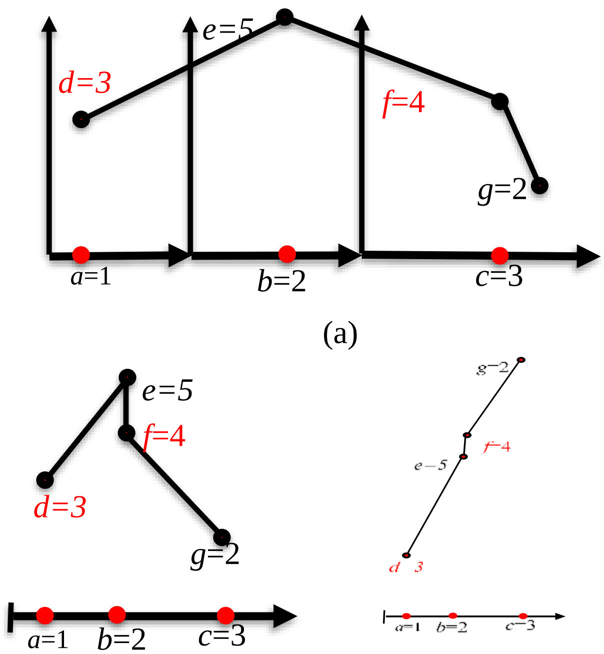
(a) Vertical simplification of Fig. 3b. (b) Minimized representation.

Fig. 4. 7-D point **x** = (*a*,*b*,*c*,*d*,e,*f*,*g*) = (1,2,3,5,3,4,2) in ILC2 with vertical sides.

Also, a more generic simplified notation is used in this figure with attributes named from *a* to *g*, because any of the coordinates $\{x_i\}$ can be assigned to be on the baseline or on link lines and in any order. This figure allows a full restoration of all seven values and requires for them only four nodes and three edges, while Parallel Coordinates require seven nodes and six edges.

The visual representation in Fig. 4 with vertical sides can be interpreted as follows. All vertical values *a*-*g* are located on respective coordinates A-G, which are vertically collocated on what is commonly known as Cartesian y-coordinate, while the ILC baseline occupies the Cartesian x-coordinate. Thus, ILBC in Fig. 4 is a **combination of two ILCs – horizontal and vertical**. It will denote such coordinates **ILC2**.

Next, ILC2 compares with Shifted Paired Coordinates (SPC) [3] on the same 7-D point shown in Fig. 5. In Fig. 5a SPC *d*, *e*, *f*, and *g* are also vertical, but start at the origins of individual horizontal coordinates, which are paired. SPC also requires 4 nodes and 3 edges that are longer than in ILBC in Fig. 4b. In ILC and ILBC above, the location of all coordinates on the horizontal baseline is fixed with their values located on this baseline.



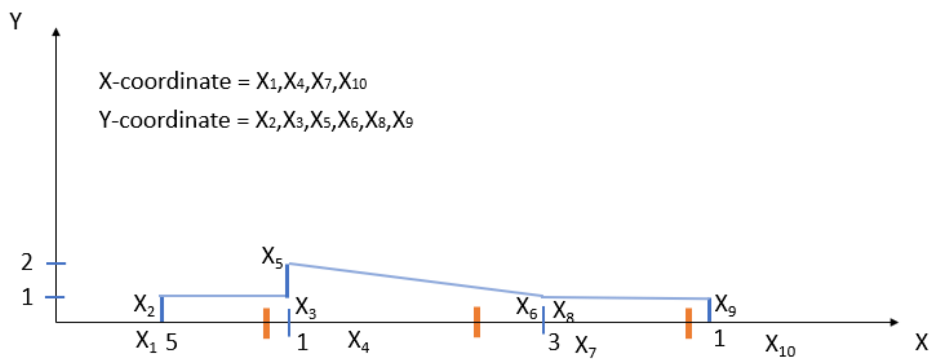
(b) (c)

Fig. 5. (a) 7-D point **x** = (*a*,*b*,*c*,*d*,e,*f*,*g*) = (1,2,3,5,3,4,2) in SPC, (b) in ILBC partial dynamic, (c) in fully dynamic.

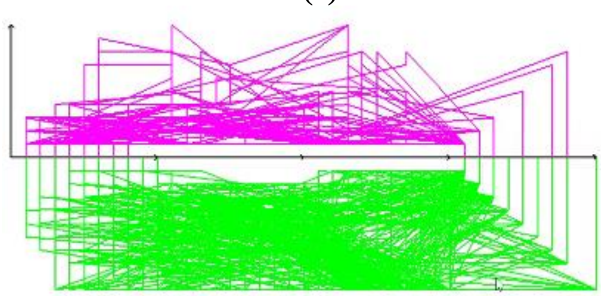
It is called a **static mapping** [3]. In the **dynamic mapping** of the given n-D point **x**, the location of the next value x_{i+1} in its 2-D graph **x*** depends on the location and value of prior x_i. It is a common concept for all General Line Coordinates [3], not only ILC and ILBC.

Fig. 5b shows the same 7-D point. Here the coordinate B starts at point *a*=2 of coordinate A with value *b*=2 located at distance two from point *a*=1. Respectively coordinate C starts at point b=2 with point *c*=3 located at the distance 3 from *b*=2. The respective vertical coordinates *d*, *e*, *f*, and *g* start at the origin of the horizontal baseline. Thus, all of them are **collocated and static**. As horizontal coordinates are dynamic, but vertical are static, therefore, ILBC is called **partial dynamic ILC2**. Fig. 5c shows a **full dynamic ILC2** where vertical coordinates are dynamic in the same ways as horizontal coordinates, where the location of *e*, *f* and *g* points depends on the location of their prior points.

Figs. 6 and 7 show WBC data of two classes ILC where vertical coordinates are collocated and horizontal are static. Figs. 6a and 7a are example of one case from WBC dataset for Figs. 6b and 7b. Drawing classes “mirrored” in Fig. 6 allows to compare and see the difference and similarities of patterns of two classes without their occlusion. Fig. 7 shows much better separation of WBC classes in fully dynamic ILC2.

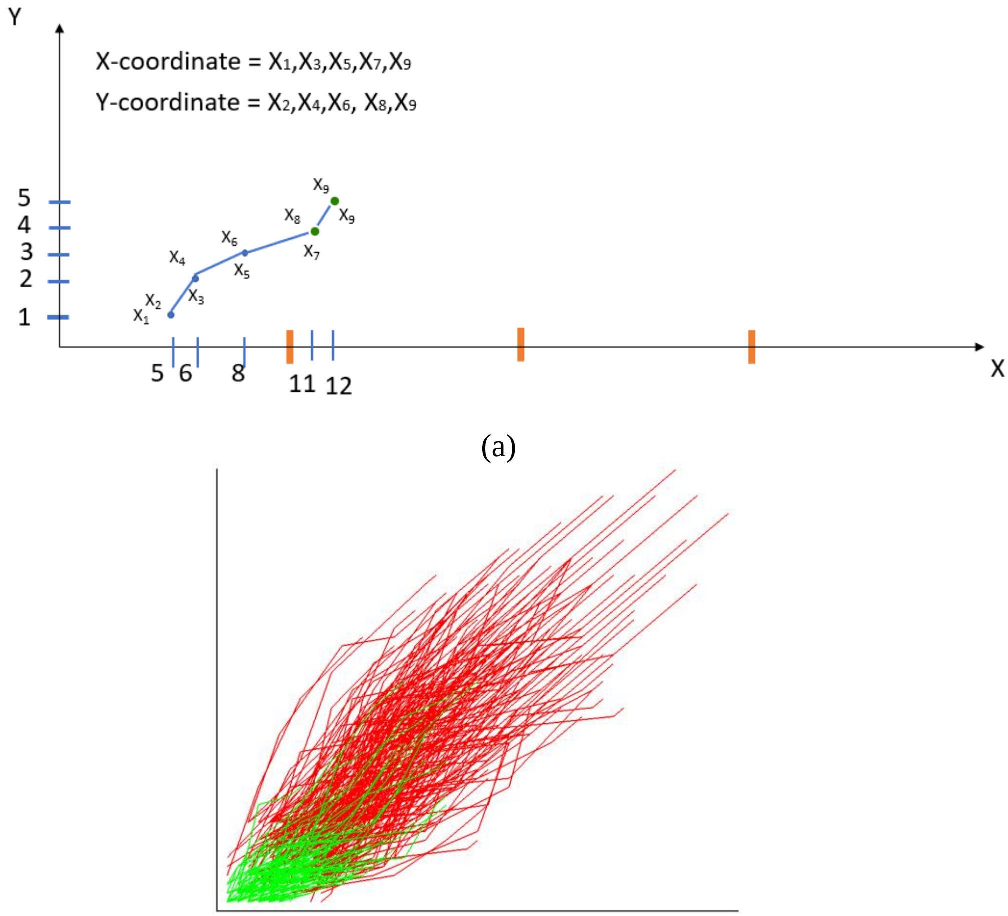


(a)



(b)

Fig. 6. (a) Example coordinates. (b) Wisconsin Breast Cancer dataset of two classes “mirrored” in partial dynamic ILBC.



(a)

(b)

Fig. 7. (a) Example coordinates.
(b) Wisconsin Breast Cancer dataset in fully dynamic ILC2.

The general formula to locate pairs (x_i, x_j) in **partial dynamic IL2** is given by mapping *L* as follows:

$L(x_i, x_j) = (x_1+x_3+\ldots+ x_i, x_j).$

Respectively the general formula to locate pairs (x_i, x_j) in **fully dynamic ILBC** is given by mapping *L* as follows:

$L(x_i, x_j) = (x_1+x_3+\ldots+ x_i, x_2+x_4+\ldots+ x_j).$

Example: In Fig. 7a, the red point value (5,1) presents x_1 and x_2. x_{3s}, x_{4s}, x_{5s}, x_{6s},

x_{7s}, x_{8s}, x_{9s}, and x_{10s} are the representation of x_3, x_4, x_5, x_6, x_7, x_8, x_9, and x_{10} in this partially dynamic In-line coordinates. The yellow point value (6,2) presents x_{3s} and x_{4s} because the position of x_{3s} depends on where x_1 is. This means that to retrieve the true value of x_3, value of x_{3s} is subtracted to the value of x_1. Therefore, $x_3 = x_{3s} - x_1 = 6 - 5 = 1$. This process repeats for all the remain points where x_5, x_7, x_9 depend on x_{3s}, x_{5s}, x_{7s}, respectively. This process also repeats for all the even attribute such as x_4, x_6, x_8 depend on x_2, x_{4s}, x_{6s}, respectively. With that, the drawn x values are $x = (5,1,6,2,8,3,11,4,12,5)$, and the true x values are $x = (5,1,1,1,2,1,3,1,1,1)$.

Next, **weighted dynamic ILBC** is introduced with is given by mapping L_w as follows:

$L_w(x_i, x_j) = (w_1x_1 + w_3x_3 + \ldots + w_ix_i, w_2x_2 + w_4x_4 + \ldots + w_jx_j)$.

where W={w_i} is a set of weights assigned to coordinates.

Example: Euclidian between two points can be used as W.

CHAPTER III

CLASSIFICATION AND REGRESSION ALGORITHMS WITH IN-LINE COORDINATES

Box Classification Algorithm

In the published paper [2], Dr. Boris Kovalerchuk and Hoang Phan introduced BC algorithm. The main idea of the BC algorithm is finding a good box with high purity and a large number of cases, and record this box, then remove all cases, which are in this box, and repeat the process of finding other good boxes in remaining cases and continue this process until all cases of all classes will be in one of the good boxes. This process is interactive and partially automated. Automation includes computing parameters of the candidates for the good boxes.

The BC algorithm operates on n-D data visualized in ILBC in the following major steps.

Step 1: Search/discover "good" boxes B_i in these visualizations that cover dataset cases as pure as possible. Good box' criterion is usually decided by number of cases that it covers and its purity. For instance, the exhausted grid search is used to discover boxes with WBC dataset. With WBC dataset, a "good" box is discovered when it covers at least more than 10% cases of the remaining data. The search of boxes and rules is a sequential hierarchical process for each class. If there are multiple "good" boxes, then these "good" boxes ranking is based on its purity. *Purity* of the box is the number of cases of each class in the box the ratio of the cases of the dominant class to the number of cases of all other classes in the box. Here it is assumed that the graph **x*** of n-D point **x** crosses box B. In the rule below, it will be denoted $\mathbf{x} \in B$ for short. This definition leads to simpler interpretation of rules based on such boxes, rather than an alternative definition which requires that only an edge to graph **x*** crosses the box B_i.

Step 2: Form basic BC rules with the discovered boxes:

$$R_i: \text{if } \mathbf{x} \in B_i \Rightarrow \mathbf{x} \in \text{Class } C_i. \quad (3.1)$$

For WBC dataset, there are 13 rules as presented in table 2.

The general format of the rules is:

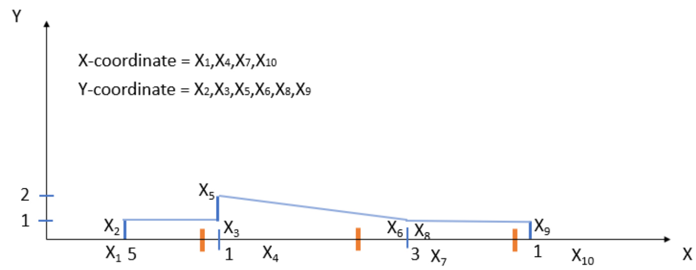
$$R_i: \text{if } \mathbf{x} \in B_i \;\&\; \mathbf{x} \notin (B_m \cup B_p \cup \ldots \cup B_t) \Rightarrow \mathbf{x} \in \text{Class } C_i. \quad (3.2)$$

Here B_i is a current “good” box, and other boxes are prior “good” boxes with cases from these boxes removed before B_i is searched. B_m, B_p, and B_t are other boxes.

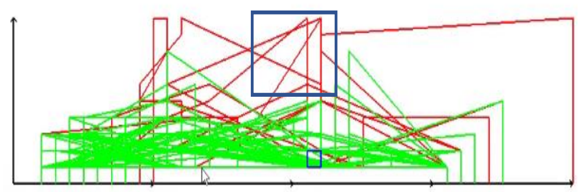
Step 3: Test BC rules on independent data.

Step 4: Pruning a set of discovered rules to decrease overfitting.

The output of *Iterative Visual Logical Classifier* algorithm results in series of rectangular areas. Fig. 8a shows one case and Fig. 8b shows more cases from WBC visualized by the IVLC algorithm.



(a)



(b)

Fig. 8. (a) Example coordinates.
(b) Examples of boxes discovered in Wisconsin Breast Cancer dataset.

The main steps of discovering boxes are:

Step 1.1. Create a *grid* in the ILC area. Each cell of the *grid* is a *box*. With WBC dataset box size is increased vertically and then horizontally after the full grids is search for each box size.

Step 1.2. If the number of boxes (grid cells) and the number of n-D points is relatively small compute purity of *each* box. For a large number of boxes and n-D points use optimization and heuristic algorithms such as genetics algorithms.

Step 1.3. For each class create a list of boxes where this class dominates. All boxes in this list must cover more cases than the desired percentage of the remaining cases. For instance, WBC dataset boxes must cover more than 10% cases of the remaining data.

Step 1.4. For a given class pick up a box with highest ranking. Box ranking is based on its purity. Highest ranking box is used to classify the remaining cases which have not been classified.

The main steps of forming basic BC rules with the discovered boxes:

Step 2.1. Create a classification rule with highest ranking box accordance with (3.1). Example: for WBC dataset, there are 382 green cases and no red cases that crosses box B_1 (Fig. 10). Therefore, rule R_1 is created as follow,

$$R_1: \mathbf{x} \in B_1 \Rightarrow \mathbf{x} \in G \text{ (Benign, 382 cases).} \quad (3.1.1)$$

More boxes and rules are identified later in table 1.

Step 2.2. Exclude all cases that are in these boxes.

Step 2.3. Conduct step 1.1 to step 2.2 for all remaining cases.

The pruning step is to deal with many “mini” boxes that contain fewer than a certain percentage of the total original dataset with low level of generalization to avoid overfitting and data memorization. With Wisconsin Breast Cancer dataset, it was decided intuitively that if a box classified fewer than 17 cases of the total number of cases which is 683 then it is very likely that this box is overfitting the data. This problem is also known for decision trees. Without control the depth of decision tree and the number of cases in each terminal node would lead to many terminal nodes with only few cases in each of them. The pruning of decision trees removes overfitting but decreases the accuracy of classification.

Example: box B_5 classified only 14 red cases (about 2.05% of 683 cases). This is too few cases, so box B_5 is likely overfitted WBC data.

The step 4 of the BC algorithm employs a version of this pruning approach which: (a) associates “mini” boxes with the larger boxes interactively or (b) refuses to predict cases that belong to “mini” boxes. The association (a) is conducted as follows. Consider two boxes B_1 and B_2 for class C_1 the visualization allows to see their mutual location and to create a joint rule based on them. If boxes are adjacent a single bigger box $B_{1,2}$ is produced from them. If the boxes are not adjacent that a new rule is formed:

$$\text{If } \mathbf{x} \in B_1 \text{ or } \mathbf{x} \in B_2 \text{ then } \mathbf{x} \in C_1$$

The general for of a new rule is,

$$\text{If } \mathbf{x} \in B_1 \cup B_2 \text{ then } \mathbf{x} \in C_1$$

The interactivity of the BC algorithm has an advantage of allowing the end-users to observe “mini” boxes and decide to follow (a) or (b),

$$R_1: \mathbf{x} \in B_1 \Rightarrow \mathbf{x} \in G \text{ (Benign, 382 cases).}$$

Linear Classification and Regression Algorithm.

Design of rules based on boxes have limitations. One of them is locality of each box, i.e., the box covers only cases that are in this box. Typically, several boxes are needed to cover all data, while a single linear classifier can cover all data if data are linearly separable. The goal of this section is proposing an analog of linear classifier in ILC. Fig. 9 illustrates the proposed approach. First, a black line is built which is used to project all cases of both classes to this line. If the projected endpoints of cases of one class C mostly concentrate on the one side of the discrimination blue line, then a linear discrimination model M is discovered where *T* is threshold on the black line shown with the blue discrimination line:

$$M(\mathbf{x}) > T \Rightarrow \mathbf{x} \in C \tag{3.3}$$

Another more common linear discrimination model for two classes C and Q is:

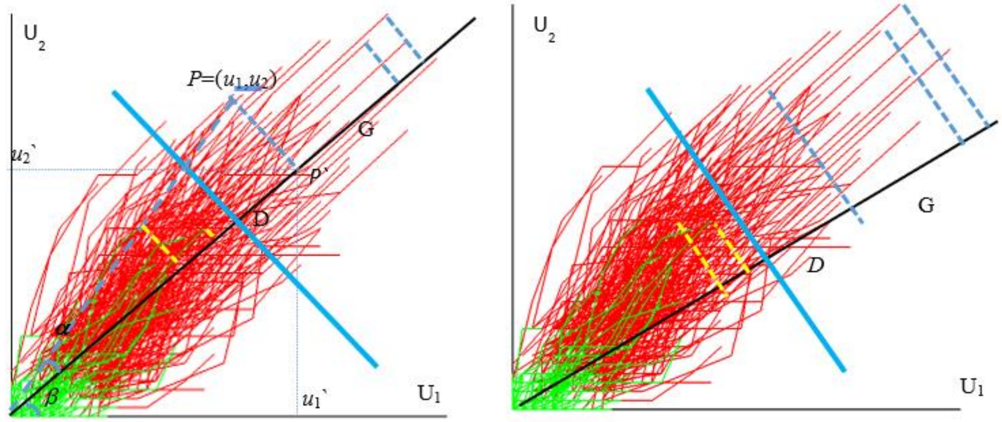
$$M(\mathbf{x}) > T \Rightarrow \mathbf{x} \in C \text{ else } \mathbf{x} \in Q \tag{3.4}$$

The model (3.3) only covers n-D points **x** where $M(\mathbf{x}) > \mathrm{T}$. This is the situation in Fig. 11 for red class above the blue line. This means that if a single model (3.4) cannot be built, several models like (3.3) need to be built that may require different black lines, where endpoints are projected as shown in Fig. 9b. Moreover, it can relax a requirement that only endpoints are projected. It can project some intermediate nodes $\boldsymbol{x}_k$ and $\boldsymbol{x}_u$ of graph of **x*** for $k < n$ and $u < n$, where n is the dimension of n-D point **x** getting models like,

$$M(\mathbf{x}_k) > T \Rightarrow \mathbf{x} \in C. \tag{3.5}$$

$$M_1(\mathbf{x}_k) > T_k \;\&\; M_2(\mathbf{x}_u) > T_u \Rightarrow \mathbf{x} \in C. \tag{3.6}$$

Those intermediate points can be found by the BC algorithm presented above. Below an ILC classification and regression algorithms are presented. Figs. 9a and 9b look different because the black lines are drawn differently, which allow to optimize the prediction parameters and accuracy. The endpoints of the black lines are chosen so that a higher precision accuracy can be obtained.



(a) (b)

Fig. 9. Linear Classification and Regression algorithm with projection line at different angles in full dynamic ILP2.

CHAPTER IV

CASE STUDY FOR BOX CLASSIFICATION ALGORITHM IN PARTIAL DYNAMIC ILC2.

This section presents the results of the computational experiment for discovering classification rules for WBC data encoded in partial dynamic ILC2 using the BC algorithm. The discovered 13 pure boxes are presented in table 1. These rules can be simplified and pruned interactively. With WBC dataset, exhaustive grid search is used to discover the hyper-parameters in table 1 (column 2 and 5). Exhaustive grid search can be used with WBC dataset because WBC dataset has a low-resolution attribute with only ten values in each attribute. With higher dimension and high-resolution dataset, exhausted grid search will take much greater computational time because there will be much more cells in the grid. We avoided such computations by proposing a heuristic algorithm that is described in section V. The boxes cover all WBC cases. Values x_1, x_2, y_1, y_2 identify left, right, bottom, and top corners of the box within ILC2 display.

Table 1. Discovered boxes ILC2.

Box	x_1, x_2, y_1, y_2	Cases	Box	x_1, x_2, y_1, y_2	Cases
B_1	15,20.5,1,1.5	382	B_2	23.5,39.5,8.5,10	166
B_3	1,3.5,0.5,2	28	B_4	20,22.5,6,6.5	26
B_5	9.5,10,5,6.5	14	B_6	16,21,0.5,2	18
B_7	17.5,18.5,3,3.5	23	B_8	14.5,17,2.5,3	7
B_9	28.5,29,2.5,3.5	4	B_{10}	17.5,18.5,3,3.5	10
B_{11}	14.5,15,5.5,6	4	B_{12}	26.5,27,7,7.5	1
B_{13}	28,28.5,0.5,9.5	10			

Table 2 presents the rules constructed from these boxes in the hierarchical process of the BC algorithm that was described above. The benign class is denoted as B and is drawn as green with letter G used to identify this class in table 2. Respectively, the class malignant is denoted by M and R (red) for short. These rules are created using the same concept describe in example (3.1.1).

Table 2. Rules R_1-R_{13} with precision P=100%.

Benign, B (green, G) class rules.
R_1: $\mathbf{x} \in B_1 \Rightarrow \mathbf{x} \in G$ (382 cases)
R_3: $\mathbf{x} \in B_3 \Rightarrow \mathbf{x} \in G$ (28 cases)
R_6: $\mathbf{x} \in B_6$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \Rightarrow \mathbf{x} \in G$ (18 cases)
R_8: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \cup B_7 \cup B_{10} \cup B_{13} \Rightarrow \mathbf{x} \in G$ (7 cases)
R_9: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \cup B_7 \cup B_{10} \cup B_{13} \Rightarrow \mathbf{x} \in G$ (4 cases)
R_{11}: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \Rightarrow \mathbf{x} \in G$ (4 cases)
R_{12}: $\mathbf{x} \in B_{12}$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \cup B_7 \cup B_{10} \Rightarrow \mathbf{x} \in G$ (1 case)
Malignant, M (red, R) class rules.
R_2: $\mathbf{x} \in B_2 \Rightarrow \mathbf{x} \in R$ (166 cases)
R_4: $\mathbf{x} \in B_4 \Rightarrow \mathbf{x} \in R$ (26 cases)
R_5: $\mathbf{x} \in B_5$ & $\mathbf{x} \notin B_1 \cup B_3 \Rightarrow \mathbf{x} \in R$ (14 cases)
R_7: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_1 \cup B_3 \cup B_6 \Rightarrow \mathbf{x} \in R$ (13 cases)
R_{10}: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_3 \cup B_6 \cup B_8 \cup B_9 \Rightarrow \mathbf{x} \in R$ (10 cases)
R_{13}: $\mathbf{x} \in B_{13}$ & $\mathbf{x} \notin B_1 \cup B_3 \cup B_6 \cup B_8 \cup B_9 \cup B_{11} \cup B_{12} \Rightarrow \mathbf{x} \in R$ (10 cases)

The boxes and rules in tables 1 and 2 cover all 444 B cases and 239 M. Boxes B_1-B_4 and respective rules R_1-R_4) cover most of the cases (602 cases) with 100% precision without any misclassified cases. This means that 88.14% of all cases are classified by simplest single box rules without any other boxes involved. The other rules involve "negated" boxes requiring that the case does not belong to these boxes to satisfy the rule. Some rules in table 2 such as rule R_8 have simplified forms too with reduced "negated" boxes, because all cases of some boxes are covered by other boxes in these rules.

Table 2 shows that class G has more rules/boxes with smaller coverage (four rules that cover from one to seven cases with total 16 cases covered by these rules). Boxes can be called "*mini*" boxes. In contrast, class R has no rules and boxes with such small coverage. Its four rules with smaller coverage include from 10 to 14 cases with 47 total cases. It means a better generalization for R class, than for G class, in these rules. When rules are analyzed with large coverage, the situation is the opposite.

The first three G rules cover 428 cases (96.4% of G cases), while the first two R rules (rules R_2 and R_4) cover 192 cases, 80.3% of R cases). Next, the rules that cover a small number of cases are more complex. They include two to seven “negated” boxes. This is rather a memorization of cases, than their generalization.

Such complex rules are needed for a small number of cases. Domain experts correctly captured/engineered a few critical attributes. This can indicate superior human abilities to generate informative features manually. While, deep learning algorithms can automatically discover informative features, often it is challenging to interpret and explain them.

A comparable Decision Tree (DT) built on the 90% of the same data is %97.40 accurate too but has multiple terminal nodes with few cases in each of them as presented in the appendix B. It is rather overfitting and data memorization. This tree has a total of 35 nodes, 29 terminal nodes with 10 of these terminal nodes contain seven or fewer cases [16]. In contrast the BC algorithm produced 13 boxes/rules, and only four of them contain seven or fewer cases on all WBC data.

Figs. 10-16 illustrate all boxes showing the cases, which cross these boxes. In addition, Figs. 12-16 show also the cases of other colors, which are removed before discovering a given box by requiring not to belong to a set of prior boxes, listed in the rules in table 2.

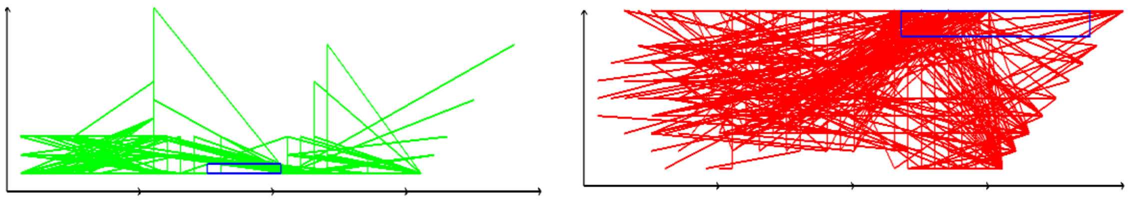
Example: In Fig. 10, 382 green cases is classified with box B_1.

The BC algorithm removed theses 382 green cases before the next search. Because BC algorithm is sequential hierarchical process, it searches for each box in the sequential order such as B_1, then B_2 and then B_3 and so on. This means that after box B_1, there will be 62 green cases remaining out of 444 total green cases for the next search. Because of the small number of cases left, there is a good chance that a rule for them can be overfitting. Therefore, it will be beneficial to analyze the number of green cases classified without removing cases of the prior boxes to get another rule with higher coverage. The BC algorithm precision can be different depending on hierarchical order. The BC algorithm hierarchical order can be changed depend on how boxes' criteria is decided by user.

Figs. 10-11 represent situations when boxes are discovered without preconditions on other boxes. These pictures show boxes B_1, B_2, B_3, and B_4 with only cases that cross them. Figs. 12-16 represent situations with boxes discovered with a precondition that cases that cross prior boxes. In Fig. 17, this graph shows the data of the patient #1000025 on fully dynamic In-line Coordinates. The X-coordinate represents the value of x_1, x_4, x_7, and x_{10}. The y-coordinate represents the value of x_2, x_3, x_5, x_6, x_8, and x_9. All attribute of patient #100025 is demonstrated in Fig. 17, such as x_1=5, x_2=1, x_3=1, x_4=1, x_5=2, x_6=1, x_7=3, x_8=1, x_9=1, and x_{10}=1. All patients in WBC dataset are represented in this manner.

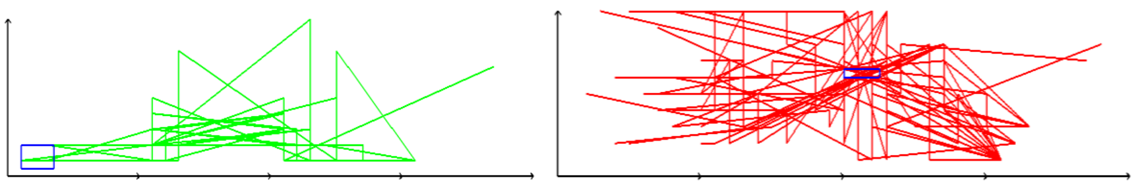
In these figures for each box B_i, the first picture shows only cases which cross box B_i after removing cases from B_1-B_4 and the second picture also shows cases from the opposite class that cross B_i without removing cases that cross prior boxes. For example,

Fig. 12 first shows 14 red cases that cross box B_5 and then it shows both read and green cases without removing 382 cases that cross box B_1 and 28 cases that cross box B_3. For box B_6 the first picture shows 18 green cases, and the second picture shows both green and red cases without removing red cases from boxes B_2 and B_4.



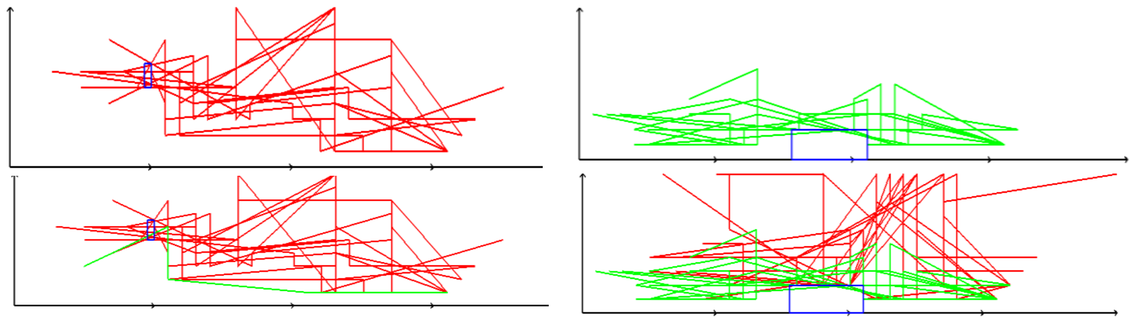
Box B_1: 382 green cases. Box B_2: 166 red cases.

Fig. 10. Boxes B_1 and B_2.



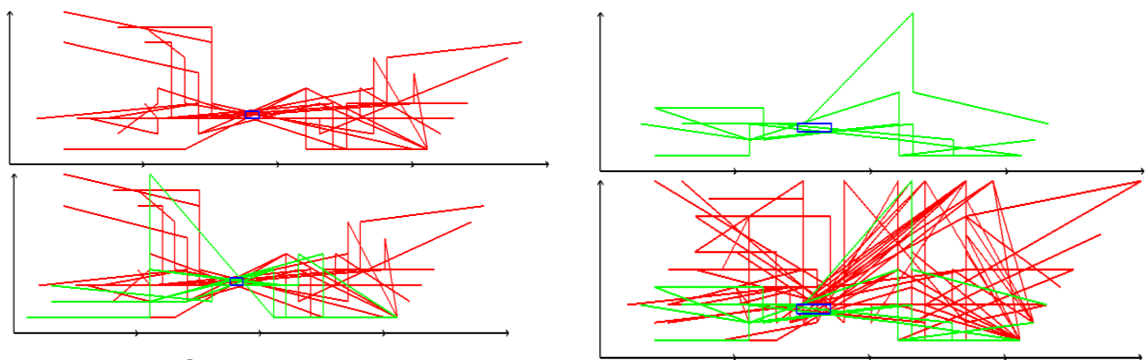
Box B_3: 28 green cases. Box B_4: 26 red cases.

Fig. 11. Boxes B_3 and B_4.



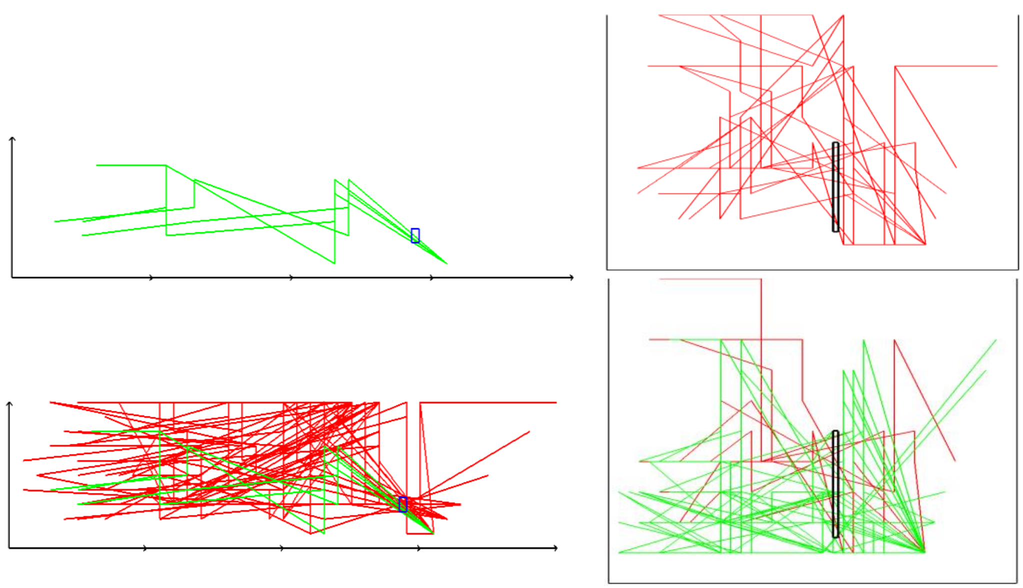
Box B_5: 14 red cases. Box B_6: 18 green cases.

Fig. 12. Boxes B_5 and B_6.



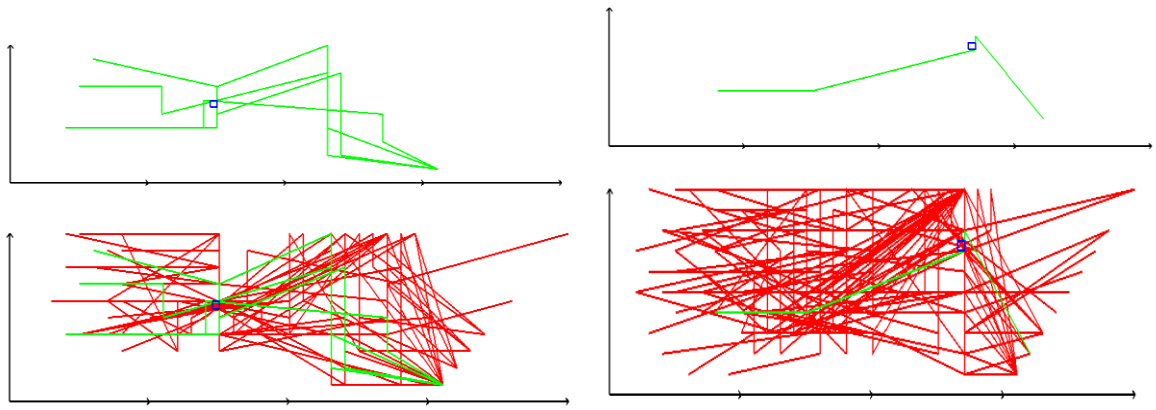
Box B_7: 23 red cases. Box B_8: 7 green cases.

Fig. 13. Boxes B_7 and B_8.



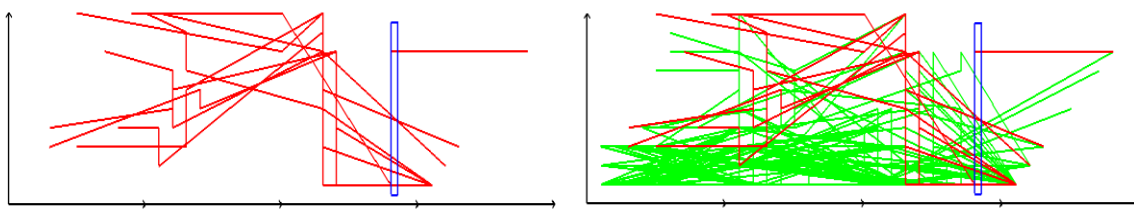
Box B_9: 4 green cases. Box B_{10}: 7 green cases.

Fig. 14. Boxes B_9 and B_{10}.



Box B_{11}: 4 green cases. Box B_{12}: 7 green cases.

Fig. 15. Boxes B_{11} and B_{12}.



Box B_{13}: 10 red cases.

Fig. 16. Box B_{13}.

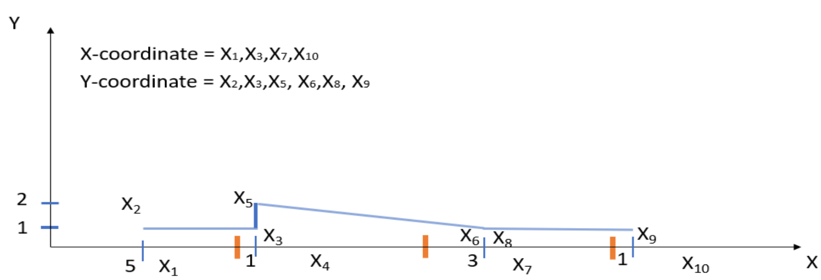


Fig. 17. Example coordinates for Figs.10-16.

Pruning. Boxes B_9, B_{11} with four green cases each and box B_{12} with one green case can be used to prune the set of rules by creating modified rules,

$$R_{9M}: \mathbf{x} \in B_9 \Rightarrow \mathbf{x} \in R \text{ (47 red /4 green).}$$

$$R_{11M}: \mathbf{x} \in B_{11} \Rightarrow \mathbf{x} \in R \text{ (28 red /4 green).}$$

R_{12M}: $\mathbf{x} \in B_{12} \Rightarrow \mathbf{x} \in R$ (52 red /1 green).

Here $\mathbf{x} \in B_i$ means that polyline for n-D point **x** crosses the box B_i.

These rules have a low error rate. Currently, the pruning process is interactive, therefore the end-users can explore them and accept if the error rate is tolerable.

Joining rules. The next task is decreasing the number of rules, that is demonstrated on 13 rules shown above. The proposed approach joins rules by combining them including the use of else condition. In contrast with the pruning, this process does not introduce any error. The result is shown in table 3.

The steps of the **Rule Joining (RJ) algorithm** are:

Step 1: Combine rules with a single rectangle of a given class.

Example: $R_{1,3}$: $\mathbf{x} \in B_1 \cup B_3 \Rightarrow \mathbf{x} \in G$ (410 cases).

Step 2: Find rules in the opposite class that are conditioned by rectangles used in Step 1.

Example: Rule 5 is created after BC algorithm have removed 602 samples (410 green cases and 192 red cases) from boxes B_1, B_2, B_3, and B_4. Without removing these 602 samples, some green samples crossing through box B_5. Therefore, rule R_5 is conditioned by $\mathbf{x} \notin B_1 \cup B_3$.

Step 3: Combine rules from Steps 1 and 2.

Example: $R_{1,3,5}$: $\mathbf{x} \in B_1 \cup B_3 \Rightarrow \mathbf{x} \in G$ (else $\mathbf{x} \in B_5 \Rightarrow x \in R$) (428 cases).

Here rule R_5 covers only 14 cases that can be viewed as a potential overfitting, while rule $R_{1,3,5}$ covers 428 cases. The else condition makes R_5 a part of the larger rule.

Table 3. Rules after joining.

Expanded benign (green, G) class rules.
$R_{1,3}$: $\mathbf{x} \in B_1 \cup B_3 \Rightarrow \mathbf{x} \in G$ (410 cases)
$R_{1,3,5}$: $\mathbf{x} \in B_1 \cup B_3 \Rightarrow \mathbf{x} \in G$ (else $\mathbf{x} \in B_5 \Rightarrow \mathbf{x} \in R$) (424 cases)
$R_{8,9}$: $\mathbf{x} \in B_8 \cup B_9$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \cup B_7 \Rightarrow \mathbf{x} \in G$ (11 cases)
$R_{11,12}$: $\mathbf{x} \in B_{11} \cup B_{12}$ & $\mathbf{x} \notin B_2 \cup B_4 \cup B_5 \Rightarrow \mathbf{x} \in G$ (5 cases)
Expanded malignant (red, R) class rules.
$R_{2,4}$: $\mathbf{x} \in B_2 \cup B_4 \Rightarrow \mathbf{x} \in R$ (192 cases)
$R_{2,4,6}$: $\mathbf{x} \in B_2 \cup B_4 \Rightarrow \mathbf{x} \in R$ (else $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in G$) (210 cases)
R_7: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_3 \cup B_6 \Rightarrow \mathbf{x} \in R$ (13 cases)
$R_{2,4,8}$: $\mathbf{x} \in B_2 \cup B_4 \Rightarrow \mathbf{x} \in R$ (else $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_5 \cup B_7 \Rightarrow \mathbf{x} \in G$) (199 cases)
$R_{2,4,6,8}$: $\mathbf{x} \in B_2 \cup B_4 \Rightarrow \mathbf{x} \in R$ (else $\mathbf{x} \in B_6 \cup B_8$ & $\mathbf{x} \notin B_5 \cup B_7 \Rightarrow \mathbf{x} \in G$) (217 cases)
R_{10}: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_3 \cup B_6 \cup B_8 \cup B_9 \Rightarrow \mathbf{x} \in R$ (10 cases)
R_{13}: $\mathbf{x} \in B_{13}$ & $\mathbf{x} \notin B_1 \cup B_3 \cup B_6 \cup B_8 \cup B_9 \cup B_{11} \cup B_{12} \Rightarrow \mathbf{x} \in R$ (10 cases)

The analysis of rules in table 3 shows that seven rules $R_{1,3,5}$, $R_{8,9}$, $R_{11,12}$, $R_{2,4,6,8}$, R_7, R_{10}, and R_{13} are equivalent to 13 original rules. Here rules $R_{8,9}$, $R_{11,12}$, R_{10}, and R_{13} cover 10 or fewer cases with total 36 cases (16 green and 20 red). Excluding these rules and, respectively refusing to classify cases that satisfy them will eliminate potential overfitting.

Model Evaluation with Worst-case *K* Fold Validation Approach.

k-fold cross validation. So far, the case study was conducted on the whole WBC dataset. What will be the accuracy of the BC algorithm in k-fold cross validation (k-fold cross validation) on these data? This question can be answer in a non-traditional way. It is an attempt to find the worst and best-case estimates for stratified 10-fold cross validation as follows. The formal concept of worst-case cross validation estimate, based on the Shannon function, was introduced in [17]. The motivation of getting the worst-case estimates is coming from the fact that k-fold cross validation only tests a *small fraction* of splits of data into training and validation sets, giving potentially an inflated average estimate, which can be misleading to life-critical applications such as cancer diagnostics.

First, consider a validation fold, which includes all 16 cases that are in “mini” boxes R_8-R_9 and R_{11}-R_{12}. These cases are likely overfitted and not generalized well by rules. Therefore, they are good candidates for the worst fold for the BC algorithm. The training data in the remaining 9 folds do not contain these cases. Thus, these “mini” boxes will not be discovered by the BC algorithm because all their cases are not in the training data. Assume that this algorithm discovered all other boxes on training data that contain 90% of all WBC with a rule,

$$R_{12M}: \mathbf{x} \in B_{12} \Rightarrow \mathbf{x} \in R \text{ (52 red /1 green).}$$

How will the BC algorithm classify cases from these “mini” boxes? There are two options: (1) refusal and (2) make an error by using modified rules like R_{12M} because R_{12M} misclassify these cases. In stratified 10-fold cross validation, a training-validation split of cases: 615 – 68. In the worst validation fold 16 cases are misclassified with 76.47% accuracy. All other folds do not have any misclassified cases (100% accuracy), because it is assumed that all other boxes are discovered. The average accuracy in all 10 folds will be average of 76.47% and 100% taken nine times – 97.65%. If BC algorithm refuses to classify these 16 cases, then the precision will be 100%. Both situations (1) and (2) can be considered as the best-case estimates.

Next, it was relaxed the assumption that all non-mini boxes B_i are discovered fully. Assume that smaller boxes $B_{i`}$ are discovered, which do not include all 52 remaining cases in the worst fold. Then the accuracy in the worst fold will be 0% with 100% accuracy in all 9 other folds with the average accuracy of 90%. This is a worst-case estimate. The average estimate will be found between the best and worst estimates.

A more detailed analysis is also possible, which involves visual analysis with possible zooming. Let us consider the largest box B_1 with 382 green cases. In stratified 10-fold cross validation, only 38 cases from this box are left for the validation fold. Fig. 10 shows the location of this box. The visual analysis allows us to identify and count lines which are at the edge of the box. If that number is fewer than 38 cases, then misclassifying 38 cases is impossible, when B_1 is learned with a subset of data.

CHAPTER V

ANALYSIS EXPERIMENTAL RESULTS AND COMPARISON WITH PUBLISHED RESULTS BOX CLASSIFICATION ALGORITHM.

5.1. Box Classification Algorithm for Wisconsin Breast Cancer Dataset with Stratified 10-Fold Cross Validation.

WBC dataset was described in chapter one and all discovered WBC rules were presented above in tables 2-3.This section analyzes rules $R_{1,3}$ and $R_{2,4}$ discovered by BC algorithm which classify above 88% of WBC dataset. To get the even number of coordinates, attribute nine was used twice. Table 4 presents results of BC algorithm on WBC data for rule $R_{1,3}$ that predicts green class based on all 444 green (benign) and 239 red (malignant) of WBC data. Table 5 presents hyper-parameters of $R_{1,3}$. WBC rule $R_{1,3}$ is:

$$R_{1,3}: \mathbf{x} \in B_1 \cup B_3 \Rightarrow \mathbf{x} \in G \text{ (Benign)}.$$

Table 4. Precision and recall of rule $R_{1,3}$ for all data.

WBC dataset	Rule precision (%)	Rule recall (%)
683 cases	100	92.34

Table 5. Hyper-parameters of the rectangles B_1 and B_3 used in rule $R_{1,3}$.

Type of data used	Hyper-parameters for B_1	Hyper-parameters for B_3
Full dataset	x_1= 15, x_2=20.5, y_1=1, y_2=1.5	x_1=1, x_2=3.5, y_1=0.5, y_2=2

Table 6. Number of cases that satisfy rule $R_{1,3}$ (green rule) with boxes B_1 and B_3(BC algorithm, WBC data) in stratified 10-fold cross validation.

90%:10% stratified random folds	Red cases correctly classified by rule R_2		Green cases misclassified as red by rule R_2	
	Training	Validation	Training	Validation
1	374	44	3	2
2	375	40	1	0
3	373	42	1	0
4	373	42	1	0
5	375	40	1	0
6	374	41	1	0
7	373	42	1	0
8	375	40	1	0
9	371	44	1	0
10	373	42	1	0
Mode	373	42	1	0

Tables 6-7 present results of stratified 10-fold cross validation of BC algorithm on WBC data for rule $R_{1,3}$ that predicts green class. Table 7 also presents the hyper-parameters that was used for each fold. Each fold contains 400 green (benign) cases and 215 red (malignant) cases from the total 444 and 239 cases, respectively.

Table 7. Precision and recall of rule $R_{1,3}$ for stratified 10-fold cross validation.

90%:10% random stratified folds	Rule precision		Rule recall		Hyper-parameters	
	Training (%)	Validation (%)	Training (%)	Validation (%)	Hyper-parameters for B_1	Hyper-parameters for B_3
1	99.20	95.65	93.5	100	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
2	99.73	100	93.75	90.91	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
3	99.73	100	93.25	95.45	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
4	99.73	100	93.25	95.45	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
5	99.73	100	93.75	90.91	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
6	99.73	100	93.25	91.11	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
7	99.73	100	93.25	95.45	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
8	99.73	100	93.75	90.91	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
9	99.73	100	92.75	100	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
10	99.73	100	93.25	95.45	x_1= 15, x_2=20.8, y_1=1, y_2=1.7	x_1=1, x_2=3.5, y_1=0.7, y_2=2.8
Average	99.67	99.57	93.38	94.56		

Tables 8-9 presents results and hyper-parameters of BC algorithm on WBC data for rule $R_{2,4}$ that predicts red class based on all WBC data without splitting to training and validation subsets. This WBC rule $R_{2,4}$ is:

$$R_{2,4}: \mathbf{x} \in B_2 \cup B_4 \Rightarrow \mathbf{x} \in R \text{ (Malignant)}.$$

Table 8. Precision and recall of rule $R_{2,4}$ for all data.

WBC dataset	Rule precision (%)	Rule recall (%)
683 cases	100	80.33

Table 9. Hyper-parameters of the rectangles B_2 and B_4 used in rule $R_{2,4}$.

Type of data used	Hyper-parameters for B_2	Hyper-parameters for B_4
Full dataset	x_1= 23.5, x_2=39.5, y_1=8.5, y_2=10	x_1=20, x_2=22.5, y_1=6, y_2=6.5

Tables 10 and 11 present results of stratified 10-fold cross validation with hyper-parameters for each fold of BC algorithm on WBC data for rule $R_{2,4}$ that predicts red class.

Table 10. Number of cases that satisfy rule $R_{2,4}$ (red rule) with boxes B_2 and B_4 (BC algorithm, WBC data) in stratified 10-fold cross validation.

90%:10% stratified random folds	Red cases correctly classified by rule R_2		Green cases misclassified as red by rule R_2	
	Training	Validation	Training	Validation
1	191	21	8	0
2	192	20	9	0
3	194	18	9	0
4	192	20	9	0
5	192	20	8	1
6	192	20	8	1
7	189	24	8	1
8	190	22	8	1
9	193	19	8	1
10	193	20	7	2
Mode	192	20	8	1

Table 11. Precision and recall rule $R_{2,4}$ for stratified 10-fold cross validation.

90%:10% random stratified folds	Rule precision		Rule recall		Hyper-parameters for B_2	Hyper-parameters for B_4
	Training (%)	Validation (%)	Training (%)	Validation (%)		
1	95.98	100	88.84	87.5	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
2	95.52	100	89.30	83.33	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
3	95.57	100	90.23	75.00	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
4	95.52	100	89.30	83.33	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
5	96.00	95.24	89.30	83.33	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
6	96.00	95.24	89.30	83.33	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
7	95.94	96	87.91	100	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
8	95.96	95.65	88.37	91.66	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
9	96.02	95	89.76	79.17	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
10	96.50	90.91	89.76	83.33	x_1= 23, x_2=39.5, y_1=8, y_2=10	x_1=16, x_2=22.5, y_1=6, y_2=6.5
Average	95.90	96.80	89.21	85.00		

5.2. Box Classification Algorithm for Page Block Classification Dataset with Stratified 10-Fold Cross Validation.

PBC dataset [15] has 5473 cases with 10 attributes each. There are 4913 cases from class Text (class C_1), 329 cases from class Horizontal Line (class C_2), 28 cases from class Graphic (class C_3), 88 cases from class Vertical Line (class C_4) and 115 cases from class Picture (class C_5). This dataset is heavily imbalanced in the number of cases of classes that range from 28 to 4913 cases.

5.2.1. Divide and Conquer Algorithm for Imbalanced PBC Data.

To classify these imbalanced data, the divide and conquer approach is used with the BC algorithm with three steps.

Step 1: Class C_2-C_5 is combined into class C_{2345}. Classify cases between class C_1 and class C_{2345}. This task is less imbalanced with 4913 cases in C_1 and 560 cases in C_{2345} than the task with 5 classes C_1-C_5.

Step 2: Classify cases between class C_2 and class C_{345} that combines classes C_3-C_5 which is also less imbalanced: 329 cases in C_2 vs 231 cases in C_{345}.

Step 3: Classify 231 cases in joint class C_{345} among three classes C_3, C_4 and C_5.

5.2.2. DT Guided (DTG) Algorithm for High-Resolution Dataset PBC Data.

PBC is a *high-resolution dataset* where each attribute has a large number of values, e.g., some attributes have 5 digits in each value. In contrast, in WBC data, each value consists of a single digit where the exhaustive grid search needs to run only on 1000 boxes. The exhaustive grid search in PBC data requires to run on a grid that is several orders of magnitude larger. So, the run time need to be decreased. In [6] a random selection of grid cells (boxes) was used to decrease search time. Here **D**T is used as a **guide** for finding promising boxes. The steps of the **DT Guided (DTG) algorithm** are as follows.

Step 1: *Build a DT* on the same data.

Step 2: Select *high purity* DT branches, where a single class highly dominates.

Step 3: *Built boxes* based on those branches (one or more boxes from the branch).

Step 4: Search for *better boxes* in the vicinity of boxes derived from the DT step (3).

The link between branches of the DT and the boxes is shown in [16]. Each branch of DT is a source of several boxes because boxes are two-dimensional, while each DT branch can have many attributes.

The *current implementation of step 3* starts from nodes of the branch that are close to the root of the DT because they typically cover more cases than the nodes which are closer to the terminal node. As an illustration consider the following branch of the DT for the 4-D point **x**= (x_1,x_2,x_3,x_4) with all $x_i \in [0,10]$,

$$\text{If } (x_1 > 5 \ \& \ x_3 > 6) \ \& \ (x_2 < 3 \ \& \ x_4 < 7) \text{ then } \mathbf{x} \in \text{ class 1.}$$

Then box B_1 from ($x_1 > 5$ & $x_3 > 6$) and box B_2 from ($x_2 < 3$ & $x_4 < 7$) are created. Respectively B_1 is defined by conditions $10 \geq x_1 > 5$, $10 \geq x_3 > 6$. For shorter notation below it writes only ($x_1>5$ & $x_3>6$) assuming that the other limit is a knows constant.

5.2.3. BC Algorithm as a Generalization of Decision Tree Algorithm.

DT and rules based on boxes serve the same goal of providing *interpretable* and *easily visualizable* models. The major limitation of models constructed by the DT algorithms is the need to select a start attribute manually (*tree root*). This *narrows the class of models* that can be discovered. It led to development of *Random Forests* (RFs) algorithms, where multiple DTs are combined by voting. RFs fundamentally expanded the class of models but with the cost of *losing interpretability*. The BC algorithm covers a *wider class of models* than DTs because they to do not require a root attribute. This is an *advantage* for the BC algorithm over the DTs.

5.2.4. ILC Box Visualization as a Richer Visualization of Decision Tree.

The description of a DT as sequences of boxes is not only an alternative way to *describe* DT but also an alternative to *visualize* it in ILC as boxes. What is the advantage of this visualization of DT relative to traditional visualization of decision trees? A DT allows tracing each n-D point in the tree, but it does not visualize and distinct those n-D points in the tree. The box visualization allows it in ILC. It shows all cases that go through the boxes and fully represents the DT. It makes this visualization richer and more informative. ILC allows distinguishing DT branches by using distinct colors when all branches are visualized together or by showing each branch in the separate ILC.

5.2.5. BC Algorithm to Decrease Overfitting.

Table 12. Weighted precision for all classes with DT for PBC dataset.

Class	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
Class 1	89.14	98.05	3950	87.41
Class 2	7.15	74.76	317	5.35
Class 3	0.00	0.00	0	0.00
Class 4	1.62	90.28	72	1.47
Class 5	2.08	46.74	92	0.97
All	100.00		4431	95.19
	Validation			
Class 1	58.10	97.56	287	56.68
Class 2	7.89	58.97	39	4.66
Class 3	0.00	0.00	0	0.00
Class 4	31.17	5.19	154	1.62
Class 5	2.83	21.43	14	0.61
All	100.00%		494	63.56
	Testing			
Class 1	87.77	98.75	481	86.68
Class 2	7.85	72.09	43	5.66
Class 3	0.00	0.00	0	0.00
Class 4	1.64	77.78	9	1.28
Class 5	2.74	60.00	15	1.64
All	100.00		548	95.26

Ideally the DT model will have a high accuracy without overfitting. In this situation, the box visualization provides a richer visualization of this DT. Much more often a DT model is not perfect but has insufficient accuracy and significant overfitting. Multiple DT branches may have terminal nodes which include just few cases rather memorizing data than learning the patterns. DT's weighted precision in table 12 demonstrates highly insufficient accuracy because there is no case from class 3 which is correctly classified.

Pruning is a common way to decrease overfitting of decision trees but at the price of decreasing accuracy. The BC algorithm allows getting higher accuracy without pruning because it is not limited by the tree structure.

5.2.6 Stratified10-Fold Cross Validation Experiment.

For stratified 10-fold cross validation, PBC dataset is split into 90%:10% where 90% used for into training and validation set and 10% for independent testing. Training and validation set was then split to 90% training set and 10% validation set.

Tables 13 to 17 present the result of BC algorithm *1st fold* in stratified 10-fold cross validation. Results for folds 2-10 are presented in appendix from table 21 to table 65.

Table 13 presents hyper-parameters of rectangles derived from the DT for this fold and table 14 presents rules based on these rectangles. This DT was build using respective 90% of training and validation set designated for this fold.

Table 13. Hyper-parameters of the rectangles B_1-B_{11} (BC algorithm, PBC dataset) of 1st fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$X_6 < 0.0011$ & $0.0015 \le X_0 < 0.0214$	B_2	$0.1550 \le X_4 < 0.9394$ & $0.0065 < X_0 \le 0.107$
B_3	$0.7525 \le X_5 < 1$	B_4	$0.0750 \le X3 < 1$ & $0.0001 \le X6 < 1$
B_5	$0 \le X3 < 0.0005$	B_6	$0.3730 \le X_4 < 1$ & $0.0115 \le X_3 < 0.537$
B_7	$0 \le X_4 < 0.12$	B_8	$0 < X_3 \le 0.0005$
B_9	$0 < X_3 \le 0.0005$	B_{10}	$0 \le X_4 \le 0.2944$
B_{11}	$0.006 \le X_2 \le 0.6058$		

Table 14. Rules R_1-R_6 (BC algorithm, PBC dataset) using boxes B_1-B_{11}.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5 \cup B_6 \cup B_7$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_8 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_8 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_8 \cup B_9 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_8 \cup B_9 \cup B_{10} \Rightarrow \mathbf{x} \in$ Class C_3

The process of classification using these rules is hierarchical like in the DTs. Case **x** is classified between class C_1 and a joined class $C_{2,3,4,5}$ and then if **x** is in $C_{2,3,4,5}$ then **x** is classified to classes C_2-C_5. See Fig. 18.

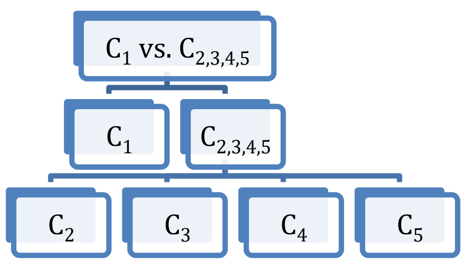


Fig. 18. Classification process.

Table 15 shows the number of cases that were predicted for training, validation, and testing sets rules in table 14. The notation C_i=>C_i indicates situations when cases of class C_i were predicted correctly as cases of class C_i, e.g., C_{2345}=>C_{2345} means that cases of the joint class C_{2345} were predicted *correctly* as cases of this joint class. The notation C_i=> C_j when $i \neq j$ indicates situations when cases of class C_i were predicted *incorrectly* as cases of class C_j.

Table 16 shows precision and recall percentages of all rules R_1-R_6 on training, validation, and testing sets.

Table 15. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 1st fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3782	55	420	5	445	5
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	359	35	42	4	48	9
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	251	2	28	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	66	6	7	0	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	8	11	1	12	0
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	2	3	1	3	0

Table 16. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 1st fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.57	95.03	98.82	95.02	98.89	90.63
R_2	91.12	79.60	91.30	80.77	84.21	84.21
R_3	99.21	94.36	100.00	93.33	91.43	96.97
R_4	91.67	92.96	100.00	87.50	87.50	77.78
R_5	91.21	90.22	91.67	100.00	100.00	100.00
R_6	91.67	100.00	75.00	100.00	100.00	100.00

The rules R_1-R_6 do not cover the same number of cases, therefore the balancing of their contribution to the **total precision** of requires using the weighted precision that will account this difference.

Example: Consider rule R_a that predicted 100 cases, which include 90 correctly classified and 10 misclassified cases (P_a=90% precision). Another rule R_b classified 200 cases, which include 160 cases correctly and 40 incorrectly classified (P_2=80% precision). Here T_a=100 and T_b=200, $W_a=T_a/(T_a+T_b)\approx 0.33$ and $W_b=T_b/(T_a+T_b)\approx 0.67$ and the weighted precision $P = 0.33 * 90\% + 0.67 * 80\% = 83.33\%$. Table 17 uses this concept of **weighted precision** P that is the weighted sum of precisions of all rules R_1-R_k:

$$P = W_1P_1 + W_2P_2 + \cdots + W_kP_k. \tag{5.1}$$

where the weight W_i of each rule R_i is computed as follows:

$$W_i = \frac{T_i}{(T_1 + T_2 \ldots\ldots T_i)}. \tag{5.2}$$

where T_i is the total number of cases which are correctly classified by rule R_i.

The process of rule generation in the BC algorithm for PBC data is based on formulas (5.1) and (5.2) for terminal level rules R_1, R_3, R_4, R_5, and R_6 which predict actual classes C_1, C_2, C_4, C_5, and C_3, respectively. The rule R_2 that is an intermediate rule, which predicts a joined class $C_{2,3,4,5}$ is excluded.

The main steps to calculated weighted precision for PBC dataset is shown below.

Step 1: Calculate weighted precision of the terminal level rules R_1, R_3, R_4, R_5, and R_6 that predict actual classes C_1, C_2, C_4, C_5, and C_3, respectively.

Step 2: Calculate weighted precision of R_1, R_3, R_4, R_5, and R_6.

Step 3: Sum up the weighted precisions from Step 2.

The results for 1st fold in stratified 10-fold cross validation are presented in table 17, where the weighted precision of all rules is 98.29% for the training set, 98.53% for the validation set, and 98.23% for the testing set.

Table 17. Weighted precision for all classes of BC algorithm for PBC dataset of 1st fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.71	98.57	3837	88.43
R_3	5.92	99.21	253	5.87
R_4	1.68	91.67	72	1.54
R_5	2.13	91.21	91	1.94
R_6	0.56	91.67	24	0.51
All	100.00		4277	98.29
	Validation			
R_1	89.29	98.82%	425	88.24
R_3	5.88	100.00%	28	5.88
R_4	1.47	100.00%	7	1.47
R_5	2.52	91.67%	12	2.31
R_6	0.84	75.00%	4	0.63
All	100.00		476	98.53
	Testing			
R_1	88.58	98.89	450	87.60
R_3	6.89	91.43	35	6.30
R_4	1.57	87.50	8	1.38
R_5	2.36	100.00	12	2.36
R_6	0.59	100.00	3	0.59
All	100.00		508	98.23

5.3. Comparison Results between Box Classification Algorithm with Published Results and Tanagra Decision Tree.

Table 18. Average precision for stratified 10-fold cross validation of BC algorithm for PBC dataset.

Fold's number	Weighted precision (%) for testing data
1st fold	98.23
2nd fold	96.65
3rd fold	98.62
4th fold	97.16
5th fold	97.84
6th fold	97.64
7th fold	96.36
8th fold	96.18
9th fold	97.16
10th fold	97.11
Average	97.30

Table 19. Comparisons with published results for PBC dataset.

Algorithm	Precision (%) on training data	Precision (%) on validation data	Precision (%) on test data	Classes with 0% precision (completely misclassified)
K-nearest Neighbor with a single 80%:20% training-validation split [18]	Not reported	93.51	Precision % on validation data	Not reported
C4-5 Decision Tree with 10-fold cross validation 90%:10% training-validation split [19]	Not reported	96.95**	Not reported	Not reported
C4-5 Decision Tree with 100% training data	96.02	N/A	N/A	**Class 3**
ID3 Decision Tree with a single 81%:9%:10% split	95.19	63.56	95.26	**Class 3**
Block Classification with 10-fold cross validation 81%:9%:10%	98.26	96.34*	97.30*	**No such classes**

*average **presumed average.

Table 20. Weighted precision for all classes with Decision Tree ID3 for PBC dataset for a single 81%:9%:10% training: validation: test split.

Class	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
Class 1	89.14	98.05	3950	87.41
Class 2	7.15	74.76	317	5.35
Class 3	**0.00**	**0.00**	**0**	0.00
Class 4	1.62	90.28	72	1.47
Class 5	2.08	**46.74**	92	0.97
All	100.00		4431	95.19
	Validation			
Class 1	58.10	97.56	287	56.68
Class 2	7.89	58.97	39	4.66
Class 3	**0.00**	**0.00**	0	0.00
Class 4	31.17	5.19	154	1.62
Class 5	2.83	21.43	14	0.61
All	100.00		494	63.56
	Testing			
Class 1	87.77	98.75	481	86.68
Class 2	7.85	72.09	43	5.66
Class 3	**0.00**	**0.00**	0	0.00
Class 4	1.64	77.78	9	1.28
Class 5	2.74	60.00	15	1.64
All	100.00		548	95.26

Table 18 shows that average precision of BC algorithm 10-fold cross validation for PBC dataset is 97.30%. Table 20 is produced as followed. Out of all data, 90% were selected randomly for training and validation and 10% for testing. Then the first 90% of data was split in 90:10 ratio for training and validation data. The precision of all classes in training is 95.19% and 95.26% in testing. Table 20 shows that precision of DT for PBC dataset is 95.26% which is slightly lower BC algorithm. In table 20, it can also be seen that the disadvantage of using DT for PBC dataset such that DT cannot classify any cases from class 3. With BC algorithm for PBC dataset, it was classified that all 5 classes with precision higher than 90% compared to DT algorithm where class 2,3,4, and 5 have precision lower than 80%. BC algorithm with In-line Coordinates also allows us to show how high dimensional dataset like PBC can be visualized on Cartesian coordinates compared to non-visualization method of DT.

The comparison of our results with published results is difficult because the authors use different splits of data. Table 19 presents results of comparisons with clearly stated what is the difference in conducted experiments. These differences are discussed in detail below. The KNN result is obtained with 80:20 split. With the KNN method, the precision is 93.51% which is lower than our BC algorithm at 97.30%. However, this lower precision is to be expected because normally it is expected to have higher precision with 90:10 split compared to 80:20 split. The C4-5 DT precision is obtained with 90:10 split in 10-fold cross validation . Therefore, C4-5 DT result would give us a better comparison with BC algorithm because PBC dataset is split in the same ratio. With BC algorithm precision is slightly higher at 97.30% compared to C4-5 DT precision at 96.95%. Table 19 also presents C4-5 DT precision with all 100% of PBC data used as training data, ID3 DT with a single 81%:9%:10% split of PBC data, and Block Classification with 10-fold cross validation 81%:9%:10% of PBC data. Both C4-5 DT and ID3 DT did not correctly classify any cases from class 3, led insufficient accuracy/ precision. The BC algorithm classified all classes with higher precision on independent test data. Furthermore, BC algorithm with In-line coordinate can visualization of PBC dataset allowing the end-user easier and faster understanding data.

CHAPTER VI

CONCLUSIONS

With In-line Coordinates data visualization, this thesis has shown the power of interpretable data classification techniques that are implemented in automatic and interactive modes with WBC dataset, and in the interactive mode with PBC dataset. The proposed BC algorithm allowed to successfully classify WBC and PBC datasets.

It was observed that BC algorithm worked well with lower feature resolution and lesser for a higher resolution and larger dataset. Higher dimension of features makes finding the best order of coordinates difficult because this process require a great amount of time. This led to further development of BC algorithm with using a Decision Tree for guidance. It allowed finding an efficient order of coordinates and a set of boxes in a practical run time. In the future, BC algorithm can be further improved by automating the process of deciding order of coordinates. BC algorithm can also be improved by multithreading exhausted grid search with computers that have more than four cores.

With In-line Based Coordinates, it was demonstrated that the power of visualization can reduce the overgeneralization of hyper-parameters produced by the guiding DT algorithm. The BC algorithm showed that it is possible to achieve better results compared to both published results of KNN and C4-5 DT algorithm. In comparison with the DT, the BC algorithm did not miss any class on highly imbalanced PBC dataset.

REFERENCES CITED

[1] Z. C. Lipton, “The mythos of model interpretability,” *Communications of the ACM*, vol. 61, no. 10, pp. 36–43, 2018.

[2] B. Kovalerchuk and H. Phan, “Full interpretable machine learning in 2D with inline coordinates,” 2021 25th International Conference Information Visualization (IV), 2021.

[3] A. Katsenou, “Kovalerchuk, B. Visual Knowledge Discovery and Machine Learning (1st edition),” *Perception*, vol. 47, no. 12, 2018.

[4] D. Dovhalets, B. Kovalerchuk, S. Vajda, and R. Andonie, “Deep learning of 2-D images representing N-D data in general line coordinates,” International Symposium on Affective Science and Engineering, vol. ISASE2018, pp. 1–6, 2018.

[5] B. Kovalerchuk, B. Agarwal, and D. C. Kall, “Solving non-image learning problems by mapping to images,” 2020 24th International Conference Information Visualization (IV), 2020.

[6] B. Kovalerchuk and A. Gharawi, “Decreasing occlusion and increasing explanation in interactive visual knowledge discovery,” Human Interface and the Management of Information. Interaction, Visualization, and Analytics, S. Yakamoto and H. Mori, Eds. pp. 505–526, 2018.

[7] R. McDonald and B. Kovalerchuk, “Lossless visual knowledge discovery in high dimensional data with elliptic paired coordinates,” 2020 24th International Conference Information Visualization (IV), pp. 286–291, 2020.

[8] C. Rudin, “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead,” *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[9] A. Inselberg, "Visual data mining with parallel coordinates." Computational Statistics 13.1, 1998.

[10] B. Kovalerchuk, "Enhancement of cross validation using hybrid visual and analytical means with Shannon function." In Beyond Traditional Probabilistic Data Processing Techniques: Interval, Fuzzy etc. Methods and Their Applications, O. Kosheleva, S. P. Shary, G. Xiang and R Zapatrin, Eds. pp. 517-543. Springer, 2020.

[11] J. Sansen, G. Richer, T. Jourde, F. Lalanne, D. Auber and R. Bourqui, "Visual exploration of large multidimensional data using parallel coordinates on big data infrastructure." In *Informatics*, vol. 4, no. 3, p. 21. Multidisciplinary Digital Publishing Institute, 2017.

[12] V. Estivill-Castro, E. Gilmore, and R. Hexel. "Constructing interpretable decision trees using parallel coordinates." International Conference on Artificial Intelligence and Soft Computing. Springer, Cham, 2020.

[13] G.K. Tam, V. Kothari, and M. Chen, “An analysis of machine-and human-analytics in classification.” *IEEE Transactions on Visualization and Computer Graphics*, 23(1), pp.71-80. 2016

[14] Y. Xu, W. Hong, N. Chen, X. Li, W. Liu, and T. Zhang, "Parallel filter: a visual classifier based on parallel coordinates and multivariate data analysis." In International Conference on Intelligent Computing, pp. 1172-1183. Springer, Berlin, Heidelberg, 2007.

[15] D. Dua and C. Graff, "UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science." http://archive. ics. uci. edu/ml (2010).

[16] B. Kovalerchuk and D. Hayes, "Discovering Interpretable Machine Learning Models in Parallel Coordinates." arXiv preprint arXiv:2106.07474, 2021.

[17] D. Zaky and P.H. Gunawan, "Computational Parallel of K-Nearest Neighbor on Page Blocks Classification Dataset." In 2020 8th International Conference on Information and Communication Technology (ICoICT), pp. 1-4. IEEE, 2020.

[18] S. Eschrich, N. V. Chawla, and L. O. Hall. "Generalization Methods in Bioinformatics." In BIOKDD, vol. 2, pp. 25-32. 2002.

APPENDIX

A.1. BC algorithm for PBC dataset of 2nd fold in stratified 10-fold cross validation.

Table 21. Hyper-parameters of the rectangles B_1-B_9 (BC algorithm, PBC dataset) of 2nd fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0 \le X_0 < 0.0005$ & $0.0015 \le X_5 < 0.0165$	B_2	$0.0002 \le X_9 < 1$
B_3	$0.0015 \le X_0 < 0.101$ & $0.4505 \le X_5 < 1$	B_4	$0.0001 \le X_6 < 0.11$ & $0.47 \le X_5 < 1$
B_5	$0 \le X_4 < 0.154$ & $0.0015 \le X_0 < 1$	B_6	$0.0125 < X_3 \le 1$
B_7	$0 < X_3 \le 0.0005$	B_8	$0 \le X_4 \le 0.2944$
B_9	$0.006 \le X_2 \le 0.6058$		

Table 22. Rules R_1-R_6 using boxes B_1-B_9 (BC algorithm, PBC dataset) of 2nd fold in stratified 10-fold cross validation.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_6 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_6 \cup B_7 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_6 \cup B_7 \cup B_8 \Rightarrow \mathbf{x} \in$ Class C_3

Table 23. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 2nd fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3887	146	432	22	466	13
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	405	23	44	5	50	11
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	249	2	30	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	65	6	7	0	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	10	11	2	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	1	3	2	3	0

Table 24. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 2nd fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	96.38	97.66	95.15	97.74	97.29	94.91
R_2	94.63	89.80	89.80	84.62	81.97	87.72
R_3	99.20	93.61	100.00	100.00	91.43	96.97
R_4	91.55	91.55	100.00	87.50	87.50	77.78
R_5	89.25	90.22	84.62	100.00	92.31	100.00
R_6	95.65	100.00	60.00	100.00	100.00	100.00

Table 25. Weighted precision for all classes of BC algorithm for PBC dataset of 2nd fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	90.20	96.38	4033	86.94
R_3	5.61	99.20	251	5.57
R_4	1.59	91.55	71	1.45
R_5	2.08	89.25	93	1.86
R_6	0.51	95.65	23	0.49
All	100.00		4471	96.31
	Validation			
R_1	89.19	95.15	454	84.87
R_3	5.89	100.00	30	5.89
R_4	1.38	100.00	7	1.38
R_5	2.55	84.62	13	2.16
R_6	0.98	60.00	5	0.59
All	100.00		509	94.89
	Testing			
R_1	89.03	97.29	479	86.62
R_3	6.51	91.43	35	5.95
R_4	1.49	87.50	8	1.30
R_5	2.42	92.31	13	2.23
R_6	0.56	100.00	3	0.56
All	100.00		538	96.65

A.2. BC algorithm for PBC dataset of 3rd fold in stratified 10-fold cross validation.

Table 26. Hyper-parameters of the rectangles B_1-B_{11} (BC algorithm, PBC dataset) of 3rd fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0 .003 \leq X_0 < 0.0330$ & $0.0021 \leq X_9 < 1$	B_2	$0 \leq X_1 < 0.023$ & $0.0005 \leq X_3 < 0.0025$
B_3	$0.63 \leq X_5 < 1$ & $0.0001 \leq X_7 < 1$	B_4	$0 \leq X_6 < 0.002$ & $0 \leq X_1 < 0.0230$
B_5	$0 \leq X_3 < 1$ & $0.0015 \leq X_0 < 1$	B_6	$0.4755 \leq X_4 < 1$
B_7	$0 \leq X_4 < 0.4755$ & $0.0012 \leq X_7 < 1$	B_8	$0. 0125 < X_3 \leq 1$
B_9	$0 < X_1 \leq 0.0045$	B_{10}	$0 \leq X_4 \leq 0.2944$
B_{11}	$0.006 \leq X_2 \leq 0.6058$		

Table 27. Rules R_1-R_6 (BC algorithm, PBC dataset) of 3rd fold in stratified 10-fold cross validation using boxes B_1-B_{11}.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_5 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_6 \cup B_7$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_5 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_8 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_8 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_8 \cup B_9 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_8 \cup B_9 \cup B_{10} \Rightarrow \mathbf{x} \in$ Class C_3

Table 28. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 3rd fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3851	37	419	5	447	2
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	373	57	46	8	53	13
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	251	1	28	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	64	4	8	1	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	11	11	1	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	2	3	1	3	0

Table 29. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 3rd fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	99.05	96.76	98.82	94.80	99.55	91.04
R_2	86.74	82.71	85.19	88.46	80.30	92.98
R_3	99.60	94.36	100.00	93.33	91.43	96.97
R_4	94.12	90.14	88.89	100.00	87.50	77.78
R_5	88.30	90.22	91.67	100.00	92.31	100.00
R_6	91.67	100.00	75.00	100.00	100.00	100.00

Table 30. Weighted precision for all classes of BC algorithm for PBC dataset of 3rd fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.88	99.05	3888	89.02
R_3	5.83	99.60	252	5.80
R_4	1.57	94.12	68	1.48
R_5	2.17	88.30	94	1.92
R_6	0.55	91.67	24	0.51
All	100.00		4326	98.73
	Validation			
R_1	88.89	98.82	424	87.84
R_3	5.87	100.00	28	5.87
R_4	1.89	88.89	9	1.68
R_5	2.52	91.67	12	2.31
R_6	0.84	75.00	4	0.63
All	100.00		477	98.32
	Testing			
R_1	88.39	99.55	449	87.99
R_3	6.89	91.43	35	6.30
R_4	1.57	87.50	8	1.38
R_5	2.56	92.31	13	2.36
R_6	0.59	100.00	3	0.59
All	100.00		508	98.62

A.3. BC algorithm for PBC dataset of 4^{th} fold in stratified 10-fold cross validation.

Table 31. Hyper-parameters of the rectangles B_1-B_9 (BC algorithm, PBC dataset) of 4^{th} fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0 \le X_6 < 0.002$ & $0 \le X_1 < 0.0230$	B_2	$0 \le X_6 < 0.0005$ & $0.0030 \le X_0 < 0.0165$
B_3	$0.0005 \le X_6 < 1$ & $0.0015 \le X_0 < 1$	B_4	$0.63 \le X_5 < 1$ & $0.0001 \le X_7 < 1$
B_5	$0 \le X_6 < 0.0005$ & $0 \le X_3 < 0.0045$	B_6	$0.0125 < X_3 \le 1$
B_7	$0 < X_1 \le 0.0045$	B_8	$0 \le X_4 \le 0.2944$
B_9	$0.0041 \le X_2 \le 0.6058$		

Table 32. Rules R_1-R_6 (BC algorithm, PBC dataset) of 4^{th} fold in stratified 10-fold cross validation using boxes B_1-B_9.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_6 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_6 \cup B_7 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_6 \cup B_7 \cup B_8 \Rightarrow \mathbf{x} \in$ Class C_3

Table 33. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 4^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3868	84	421	9	460	10
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	373	12	42	1	51	6
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	249	1	30	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	64	5	8	0	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	84	12	11	0	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	3	3	0	3	0

Table 34. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 4^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	97.87	97.19	97.91	95.25	97.87	93.69
R_2	96.88	82.71	97.67	80.77	89.47	89.47
R_3	99.60	93.61	100.00	100.00	91.43	96.97
R_4	92.75	90.14	100.00	100.00	87.50	77.78
R_5	87.50	91.30	100.00	100.00	92.31	100.00
R_6	88.00	100.00	100.00	100.00	100.00	100.00

Table 35. Weighted precision for all classes of BC algorithm for PBC dataset of 4th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.98	97.87	3952	88.07
R_3	5.69	99.60	250	5.67
R_4	1.57	92.75	69	1.46
R_5	2.19	87.50	96	1.91
R_6	0.57	88.00	25	0.50
All	100.00		4392	97.61
	Validation			
R_1	89.21	97.91	430	87.34
R_3	6.22	100.00	30	6.22
R_4	1.66	100.00	8	1.66
R_5	2.28	100.00	11	2.28
R_6	0.62	100.00	3	0.62
All	100.00		482	98.13
	Testing			
R_1	88.85	97.87	470	86.96
R_3	6.62	91.43	35	6.05
R_4	1.51	87.50	8	1.32
R_5	2.46	92.31	13	2.27
R_6	0.57	100.00	3	0.57
All	100.00		529	97.16

A.4. BC algorithm for PBC dataset of 5th fold in stratified 10-fold cross validation.

Table 36. Hyper-parameters of the rectangles B_1-B_9 (BC algorithm, PBC dataset) of 5th fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0030 \leq X_0 < 0.0330$ & $0.0021 \leq X_9 < 0.905$	B_2	$0.5260 < X_5 \leq 1$ & $0 \leq X_9 < 0.0021$
B_3	$0.0005 < X_6 \leq 1$ & $0.0015 < X_5 \leq 1$	B_4	$0.0005 \leq X_6 < 1$ & $0.0015 \leq X_5 < 1$
B_5	$0 \leq X4 < 0.12$	B_6	$0.0125 < X_3 \leq 1$
B_7	$0 < X_1 \leq 0.0045$	B_8	$0 \leq X_4 \leq 0.2944$
B_9	$0.0041 \leq X_2 \leq 0.6058$		

Table 37. Rules R_1-R_6 (BC algorithm, PBC dataset) of 5th fold in stratified 10-fold cross validation using boxes B_1-B_9.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_3 \cup B_4 \cup B_5$ & $\mathbf{x} \notin B_1 \cup B_2 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_6 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_6 \cup B_7 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_6 \cup B_7 \cup B_8 \Rightarrow \mathbf{x} \in$ Class C_3

Table 38. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 5th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3956	55	425	10	445	6
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	379	15	49	2	53	3
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	249	1	30	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	65	4	8	0	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	10	11	2	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	1	3	2	3	0

Table 39. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 5th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.63	99.40	97.70	96.15	98.67	90.63
R_2	96.19	84.04	96.08	94.23	94.64	92.98
R_3	99.60	93.61	100.00	100.00	91.43	96.97
R_4	94.20	91.55	100.00	100.00	87.50	77.78
R_5	89.25	90.22	84.62	100.00	92.31	100.00
R_6	95.65	100.00	60.00	100.00	100.00	100.00

Table 40. Weighted precision for all classes of BC algorithm for PBC dataset of 5th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	90.22	98.63	4011	88.98
R_3	5.62	99.60	250	5.60
R_4	1.55	94.20	69	1.46
R_5	2.09	89.25	93	1.87
R_6	0.52	95.65	23	0.49
All	100.00		4446	98.40
	Validation			
R_1	88.59	97.70	435	86.56
R_3	6.11	100.00	30	6.11
R_4	1.63	100.00	8	1.63
R_5	2.65	84.62	13	2.24
R_6	1.02	60.00	5	0.61
All	100.00		491	97.15
	Testing			
R_1	88.43	98.67	451	87.25
R_3	6.86	91.43	35	6.27
R_4	1.57	87.50	8	1.37
R_5	2.55	92.31	13	2.35
R_6	0.59	100.00	3	0.59
All	100.00		510	97.84

A.5. BC algorithm for PBC dataset of 6th fold in stratified 10-fold cross validation.

Table 41. Hyper-parameters of the rectangles B_1-B_{11} (BC algorithm, PBC dataset) of 6th fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0030 \le X_0 < 0.0330$ & $0.0012 \le X_7 < 0.905$	B_2	$0.0015 \le X_4 < 1$ & $0.0030 \le X_5 < 1$
B_3	$0.0015 \le X_3 < 0.101$	B_4	$0 \le X_6 < 0.0005$ & $0.101 \le X_3 < 1$
B_5	$0 \le X_6 < 0.0005$	B_6	$0.0105 \le X_3 < 1$
B_7	$0.63 \le X_5 < 1$ & $0.0015 \le X_7 < 0.905$	B_8	$0.\ 0125 < X_3 \le 1$
B_9	$0 < X_1 \le 0.0045$	B_{10}	$0 \le X_4 \le 0.2944$
B_{11}	$0.006 \le X_2 \le 0.6058$		

Table 42. Rules R_1-R_6 (BC algorithm, PBC dataset) of 6th fold in stratified 10-fold cross validation using boxes B_1-B_{11}.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5 \cup B_6 \cup B_7$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_8 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_8 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_8 \cup B_9 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_8 \cup B_9 \cup B_{10} \Rightarrow \mathbf{x} \in$ Class C_3

Table 43. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 6th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3887	45	435	10	442	7
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	379	10	46	3	50	5
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	251	3	28	0	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	65	4	8	0	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	10	11	2	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	1	3	2	3	0

Table 44. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 6th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.86	97.66	97.75	98.42	98.44	90.02
R_2	97.43	84.04	93.88	88.46	90.91	87.72
R_3	98.82	94.36	100.00	93.33	91.43	96.97
R_4	94.20	91.55	100.00	100.00	87.50	77.78
R_5	89.25	90.22	84.62	100.00	92.31	100.00
R_6	95.65	100.00	60.00	100.00	100.00	100.00

Table 45. Weighted precision for all classes of BC algorithm for PBC dataset of 6th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.96	98.86	3932	88.93
R_3	5.81	98.82	254	5.74
R_4	1.58	94.20	69	1.49
R_5	2.13	89.25	93	1.90
R_6	0.53	95.65	23	0.50
All	100.00		4371	98.56
	Validation			
R_1	89.18	97.75	445	87.17
R_3	5.61	100.00	28	5.61
R_4	1.60	100.00	8	1.60
R_5	2.61	84.62	13	2.20
R_6	1.00	60.00	5	0.60
All	100.00		499	97.19
	Testing			
R_1	88.39	98.44	449	87.01
R_3	6.89	91.43	35	6.30
R_4	1.57	87.50	8	1.38
R_5	2.56	92.31	13	2.36
R_6	0.59	100.00	3	0.59
All	100.00		508	97.64

A.6. BC algorithm for PBC dataset of 7th fold in stratified 10-fold cross validation.

Table 46. Hyper-parameters of the rectangles B_1-B_{11} (BC algorithm, PBC dataset) of 7th fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0021 \le X_9 < 1$ & $0.0030 \le X_0 < 0.0315$	B_2	$0 \le X_9 < 0.0021$ & $0.7065 \le X_5 < 1$
B_3	$0.0015 \le X_4 < 0.7155$	B_4	$0 \le X_7 < 0.0012$ & $0 \le X_0 < 0.0030$
B_5	$0.0315 \le X_0 < 1$	B_6	$0.0105 \le X_0 <1$ & $0.7155 \le X_4 < 0.905$
B_7	$0.0005 \le X_5 < 0.4955$	B_8	$0. 0125 < X_3 \le 1$
B_9	$0 < X_1 \le 0.0045$	B_{10}	$0 \le X_4 \le 0.2944$
B_{11}	$0.0041 \le X_2 \le 0.6058$		

Table 47. Rules R_1-R_6 (BC algorithm, PBC dataset) of 7th fold in stratified 10-fold cross validation using boxes B_1-B_{11}.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5 \cup B_6 \cup B_7$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_8 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_8 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_8 \cup B_9 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_8 \cup B_9 \cup B_{10} \Rightarrow \mathbf{x} \in$ Class C_3

Table 48. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 7th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3855	40	440	12	475	15
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	379	11	42	3	53	5
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	250	1	29	2	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	65	5	8	1	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	10	11	2	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	2	3	2	3	0

Table 49. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 7th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.97	96.86	97.35	99.55	96.94	96.74
R_2	97.18	84.04	93.33	80.77	91.38	92.98
R_3	99.60	93.98	93.55	96.67	91.43	96.97
R_4	92.86	91.55	88.89	100.00	87.50	77.78
R_5	89.25	90.22	84.62	100.00	92.31	100.00
R_6	91.67	100.00	60.00	100.00	100.00	100.00

Table 50. Weighted precision for all classes of BC algorithm for PBC dataset of 7th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.89	98.97	3895	88.97
R_3	5.79	99.60	251	5.77
R_4	1.62	92.86	70	1.50
R_5	2.15	89.25	93	1.92
R_6	0.55	91.67	24	0.51
All	100.00		4333	98.66
	Validation			
R_1	88.63	97.35	452	86.27
R_3	6.08	93.55	31	5.69
R_4	1.76	88.89	9	1.57
R_5	2.55	84.62	13	2.16
R_6	0.98	60.00	5	0.59
All	100.00		510	96.27
	Testing			
R_1	89.25	96.94	490	86.52
R_3	6.38	91.43	35	5.83
R_4	1.46	87.50	8	1.28
R_5	2.37	92.31	13	2.19
R_6	0.55	100.00	3	0.55
All	100.00		549	96.36

A.7. BC algorithm for PBC dataset of 8^{th} fold in stratified 10-fold cross validation.

Table 51. Hyper-parameters of the rectangles B_1-B_{11} (BC algorithm, PBC dataset) of 8^{th} fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0021 \leq X_9 < 1$ & $0.0030 \leq X_0 < 0.0315$	B_2	$0 \leq X_9 < 0.0021$ & $0.7065 \leq X_5 < 1$
B_3	$0.0015 \leq X_4 < 0.7155$	B_4	$0 \leq X_7 < 0.0012$ & $0 \leq X_0 < 0.0030$
B_5	$0.0315 \leq X_0 < 1$	B_6	$0.0105 \leq X_0 <1$ & $0.7155 \leq X_4 < 0.905$
B_7	$0.0005 \leq X_5 < 0.4955$	B_8	$0.0125 < X_3 \leq 1$
B_9	$0 < X_1 \leq 0.0045$	B_{10}	$0 \leq X_4 \leq 0.2944$
B_{11}	$0.0041 \leq X_2 \leq 0.6058$		

Table 52. Rules R_1-R_6 (BC algorithm, PBC dataset) of 8^{th} fold in stratified 10-fold cross validation using boxes B_1-B_{11}.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5 \cup B_6 \cup B_7$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_8 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_8 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_{10}$ & $\mathbf{x} \notin B_8 \cup B_9 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_{11}$ & $\mathbf{x} \notin B_8 \cup B_9 \cup B_{10} \Rightarrow \mathbf{x} \in$ Class C_3

Table 53. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 8^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3855	40	440	12	475	15
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	379	11	42	3	53	5
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	250	2	29	1	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	250	2	29	1	32	3
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	83	8	11	4	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	3	3	1	3	0

Table 54. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 8^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.97	96.86	97.35	99.55	96.94	96.74
R_2	97.18	84.04	93.33	80.77	91.38	92.98
R_3	99.21	93.98	96.67	96.67	91.43	96.97
R_4	92.75	90.14	90.00	112.50	87.50	77.78
R_5	91.21	90.22	73.33	100.00	92.31	100.00
R_6	88.00	100.00	75.00	100.00	100.00	100.00

Table 55. Weighted precision for all classes of BC algorithm for PBC dataset of 8th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	86.27	98.97	3895	85.38
R_3	5.58	99.21	252	5.54
R_4	5.58	99.21	252	5.54
R_5	2.02	91.21	91	1.84
R_6	0.55	88.00	25	0.49
All	100.00		4515	98.78
	Validation			
R_1	85.12	97.35	452	82.86
R_3	5.65	96.67	30	5.46
R_4	5.65	96.67	30	5.46
R_5	2.82	73.33	15	2.07
R_6	0.75	75.00	4	0.56
All	100.00		531	96.42
	Testing			
R_1	85.07	96.94	490	82.47
R_3	6.08	91.43	35	5.56
R_4	6.08	91.43	35	5.56
R_5	2.26	92.31	13	2.08
R_6	0.52	100.00	3	0.52
All	100.00		576	96.18

A.8. BC algorithm for PBC dataset of 9th fold in stratified 10-fold cross validation.

Table 56. Hyper-parameters of the rectangles B_1-B_9 (BC algorithm, PBC dataset) of 9th fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0005 \leq X_0 < 0.0310$ & $0.0001 \leq X_6 < 0.905$	B_2	$0 \leq X_9 < 0.0060$ & $0.7065 \leq X_5 < 1$
B_3	$0.0015 \leq X_4 < 0.7155$	B_4	$0.0105 \leq X_0 < 1$ & $0.7155 \leq X_3 < 0.905$
B_5	$0 \leq X_7 < 0.0012$ & $0 \leq X_0 < 0.0030$	B_6	$0.0125 < X_3 \leq 1$
B_7	$0 < X_1 \leq 0.0045$	B_8	$0 \leq X_4 \leq 0.2944$
B_9	$0.0041 \leq X_2 \leq 0.6058$		

Table 57. Rules R_1-R_6 (BC algorithm, PBC dataset) of 9th fold in stratified 10-fold cross validation using boxes B_1-B_9.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_4 \cup B_5$ & $\mathbf{x} \notin B_1 \cup B_2 \cup B_3 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_6 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_6 \cup B_7 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_6 \cup B_7 \cup B_8 \Rightarrow \mathbf{x} \in$ Class C_3

Table 58. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 9th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3880	35	445	35	460	10
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	388	9	44	3	48	3
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	249	2	30	1	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	65	4	8	2	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	84	10	10	2	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	4	3	0	3	0

Table 59. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 9th fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	99.11	97.49	92.71	100.68	97.87	93.69
R_2	97.73	86.03	93.62	84.62	94.12	84.21
R_3	99.20	93.61	96.77	100.00	91.43	96.97
R_4	94.20	91.55	80.00	100.00	87.50	77.78
R_5	89.36	91.30	83.33	90.91	92.31	100.00
R_6	84.62	100.00	100.00	100.00	100.00	100.00

Table 60. Weighted precision for all classes of BC algorithm for PBC dataset of 9th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.90	99.11	3915	89.09
R_3	5.76	99.20	251	5.72
R_4	1.58	94.20	69	1.49
R_5	2.16	89.36	94	1.93
R_6	0.60	84.62	26	0.51
All	100.00		4355	98.74
	Validation			
R_1	89.55	92.71	480	83.02
R_3	5.78	96.77	31	5.60
R_4	1.87	80.00	10	1.49
R_5	2.24	83.33	12	1.87
R_6	0.56	100.00	3	0.56
All	100.00		536	92.54
	Testing			
R_1	88.85	97.87	470	86.96
R_3	6.62	91.43	35	6.05
R_4	1.51	87.50	8	1.32
R_5	2.46	92.31	13	2.27
R_6	0.57	100.00	3	0.57
All	100.00		529	97.16

A.9. BC algorithm for PBC dataset of 10^{th} fold in stratified 10-fold cross validation.

Table 61. Hyper-parameters of the rectangles B_1-B_9 (BC algorithm, PBC dataset) of 10^{th} fold in stratified 10-fold cross validation.

Box	Hyper-parameters	Box	Hyper-parameters
B_1	$0.0015 \leq X_0 < 0.0180$ & $0 \leq X_6 < 0.0005$	B_2	$0.0015 \leq X_4 < 0.4545$
B_3	$0.0005 \leq X_3 < 0.0805$ & $0 \leq X_7 < 0.0012$	B_4	$0.0315 \leq X_6 < 1$ & $0 \leq X_0 < 0.0030$
B_5	$0.0315 \leq X_9 < 1$	B_6	$0.\ 0125 < X_3 \leq 1$
B_7	$0 < X_1 \leq 0.0045$	B_8	$0 \leq X_4 \leq 0.2944$
B_9	$0.0041 \leq X_2 \leq 0.6058$		

Table 62. Rules R_1-R_6 (BC algorithm, PBC dataset) of 10^{th} fold in stratified 10-fold cross validation using boxes B_1-B_9.

Class C_1 rule	R_1: $\mathbf{x} \in B_1 \cup B_2 \Rightarrow \mathbf{x} \in$ Class C_1
Class $C_{2,3,4,5}$ rule	R_2: $\mathbf{x} \in B_3 \cup B_4 \cup B_5$ & $\mathbf{x} \notin B_1 \cup B_2 \Rightarrow \mathbf{x} \in$ Class $C_{2,3,4,5}$
Class C_2 rule	R_3: $\mathbf{x} \in B_6 \Rightarrow \mathbf{x} \in$ Class C_2
Class C_4 rule	R_4: $\mathbf{x} \in B_7$ & $\mathbf{x} \notin B_6 \Rightarrow \mathbf{x} \in$ Class C_4
Class C_5 rule	R_5: $\mathbf{x} \in B_8$ & $\mathbf{x} \notin B_6 \cup B_7 \Rightarrow \mathbf{x} \in$ Class C_5
Class C_3 rule	R_6: $\mathbf{x} \in B_9$ & $\mathbf{x} \notin B_6 \cup B_7 \cup B_8 \Rightarrow \mathbf{x} \in$ Class C_3

Table 63. Number of cases that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 10^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
R_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1	C_1=> C_1	C_{2345}=> C_1
	3879	40	447	30	450	10
R_2	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}	C_{2345}=> C_{2345}	C_1=> C_{2345}
	388	9	44	3	48	3
R_3	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2	C_2=> C_2	C_{345}=> C_2
	250	2	29	1	32	3
R_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4	C_4=> C_4	C_{235}=> C_4
	64	4	9	2	7	1
R_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5	C_5=> C_5	C_{234}=> C_5
	85	11	9	1	12	1
R_6	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3	C_3=> C_3	C_{245}=> C_3
	22	4	3	0	3	0

Table 64. Precision and recall that satisfy rules R_1-R_6 (BC algorithm, PBC dataset) of 10^{th} fold in stratified 10-fold cross validation.

Rule	Training		Validation		Testing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
R_1	98.98	97.46	93.71	101.13	97.83	91.65
R_2	97.73	86.03	93.62	84.62	94.12	84.21
R_3	99.21	93.98	96.67	96.67	91.43	96.97
R_4	94.12	90.14	81.82	112.50	87.50	77.78
R_5	88.54	92.39	90.00	81.82	92.31	100.00
R_6	84.62	100.00	100.00	100.00	100.00	100.00

Table 65. Weighted precision for all classes of BC algorithm for PBC dataset of 10th fold in stratified 10-fold cross validation.

Rule	Weight (%)	Precision (%)	Classified cases	Weighted precision (%)
	Training			
R_1	89.86	98.98	3919	88.95
R_3	5.78	99.21	252	5.73
R_4	1.56	94.12	68	1.47
R_5	2.20	88.54	96	1.95
R_6	0.60	84.62	26	0.50
All	100.00		4361	98.60
	Validation			
R_1	89.83	93.71	477	84.18
R_3	5.65	96.67	30	5.46
R_4	2.07	81.82	11	1.69
R_5	1.88	90.00	10	1.69
R_6	0.56	100.00	3	0.56
All	100.00		531	93.60
	Testing			
R_1	88.63	97.83	460	86.71
R_3	6.74	91.43	35	6.17
R_4	1.54	87.50	8	1.35
R_5	2.50	92.31	13	2.31
R_6	0.58	100.00	3	0.58
All	100.00		519	97.11