IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 7, 2021, accepted June 8, 2021, date of publication June 14, 2021, date of current version July 6, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3088901

Guaranteeing Correctness of Machine Learning Based Decision Making at Higher Educational Institutions

MUHAMMAD NAUMAN[1], NADEEM AKHTAR[1], ADI ALHUDHAIF[2], AND ABDULRAHMAN ALOTHAIM[3]

[1]Faculty of Computing, The Islamia University of Bahawalpur, Bahawalpur 63100, Pakistan
[2]Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
[3]Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia

Corresponding author: Muhammad Nauman (nauman@iub.edu.pk)

This work was supported by the Deanship of Scientific Research at Prince Sattam Bin Abdulaziz University, Al-Kharj, Saudi Arabia.

This work did not involve human subjects or animals in its research.

ABSTRACT State-of-the-art software technologies have enabled Higher Education Institutions to record and store large amounts of student data. Analyzing this large amount of data can facilitate the decision-making process. Decisions made in higher education institutions affect policies, strategies and actions that improve education quality. It can be argued that machine learning algorithms demonstrate a remarkable ability to recognize models and predict results based on data. However, machine learning outcomes were subject to bias and erroneous labelling. Consequently, to mitigate the impact of such errors in decision making, it is necessary to put in place mechanisms to correct decision rules. This article presents an approach to learn decision rules through supervised machine learning and proves the correctness of these rules with hierarchical Coloured Petri Nets. The use of formalism in the proposed methodology ensures that the decision rules are correct and comprehensible. Empirical results show that we improved correctness with 98.68% accuracy on decision rules. This research work contributes to the improvement of the decision-making process for the academic administration of Higher Education Institutes.

INDEX TERMS Decision making, supervised machine learning, decision trees, formal verification, formal methods, formal modeling, model checking, coloured petri nets.

I. INTRODUCTION

Modern software technologies have leveraged Higher Educational Institutions (HEIs) to store and process large amounts of educational data to even at the smallest granularity like student daily attendance records. Data storage is not sufficient for administrators and managers to make decisions. Education data, both systematically and manually stored, should be analysed to provide an adequate presentation of relevant information to support these complex processes [1]–[3].

The analysis of this large amount of stored data is crucial to support academic decision-making at institutes. The Information and Communication Technologies (ICT) at institutes process the data to support the decision-making process. Some notable examples include Accounting Systems [4], Enterprise Resource Planning [5], and academic management [6]. The decisions of HEIs include administrative or academic nature. Although there is significant progress in modern algorithms' speed and efficiency for information processing, more efficient and user-friendly information processing systems are needed to enable modern-day decision-making processes.

Machine learning (ML) algorithms, specifically Decision Trees (DT), leverage the power of extracting decision rules from large amounts of available data. Notably, machine learning algorithms show outstanding capacity for pattern recognition and predicting outcomes from data. The ability to produce models that can incorporate decision-making is their biggest strength [7].

Decision trees, a supervised machine learning algorithm, is among the best candidates to apply on educational data sets for student activities and outcomes classification. A decision tree [8], [9] is a mathematical decision support tool that represents the decisions in a tree structure. In literature, the decision tree is investigated in educational data analyses [1], [10],

The associate editor coordinating the review of this manuscript and approving it for publication was Shagufta Henna.

medical [11], protein secondary structure prediction [12], internet of things [13], electrical systems monitoring [14], and sentiment analysis [15], [16]. In many cases, it has been extended to deal with data uncertainty [17].

Arguably, the machine learning algorithms have been prone to bias, mislabelling, and technical debt [18]–[22]. Therefore, to mitigate such errors, there is a need for mechanisms to prove the correctness of decision outcomes. Formal verification is the process of establishing the correctness, safety, and liveness properties of complex software and hardware systems. Hence, formalism is a good candidate to verify and validate machine learning outcomes [23]. Therefore, formal verification can mitigate classification errors and provide a correctness guarantee of these classification outcomes [24]. The formalism allows more rigorous decision-making rules since formal models have a solid mathematical background to guarantee the correctness and comprehensibility of these rules.

Coloured Petri Nets (CP-Nets) [25] is a type of Petri-Nets [26] used in the modelling of systems that contain discrete, concurrent, and scattered events. Specifically, CP-Nets have strong mathematical logic associated with Standard ML [27] programming language. They help model both non-deterministic and stochastic processes and allow an exhaustive systematic exploration of mathematical models to prove, disprove, and guarantee correctness. These characteristics make Petri-Nets a suitable formalism to model and analyse machine learning outcomes.

Many studies have used supervised machine learning algorithms to analyse educational data; our research differs from the existing literature. (1) The data come from the academic and demographic details of the students. (2) More diverse and numerous features of data collection are included in the analysis to achieve higher overall accuracy. (3) The use of formalism has not been exploited before for guaranteeing the correctness of machine learning outcomes.

This article addressed improved academic decision-making in higher education institutes to facilitate HEIs administrators and managers. The decision-making is aided with decision rules which are derived using supervised machine learning. The correctness of decision rules is verified using Coloured-Petri-Net formalism for mitigating the classification errors and biasing. The proposed approach's effectiveness is measured using specificity, sensitivity, and accuracy analysis of decision trees and CP-Nets simulation outcomes.

The current study leverages the HEIs decision-making authorities to analyse the admission application data department-wise. The decision-maker can analyse decision rules to make effective strategic decisions for increasing or reducing student intakes to meet a particular department's capacity. It will provide insights for creating new program offerings as well as resource optimizations for departments. The decision-making rules can improve the financial resources by maximizing the use of available resources. These decision rules are beneficial to HEI decision-makers for the betterment of education policies. These research results will help society by improving educational decision-making to maximize the education facilitation to students. In order to better generalize the findings, a data set was analysed for six different departments of a higher education institution.

The main contributions of this research work are:

1) We use supervised machine learning, specifically decision trees, for analyzing educational data sets to leverage improved decision making for Higher Educational Institutes.
2) We propose the use of CP-Nets formalism to mitigate classification errors and to prove the correctness of decision rules.
3) We perform extensive experiments on the student data set to illustrate the effectiveness of CP-Net formalism implementation for guaranteeing the correctness of derived decision rules.

The paper is organized as follows. First section II presents the background and mathematical preliminaries. Then, the literature review is discussed in section III. Afterwards, section IV discusses the materials and methods, and finally the conclusion is presented in section V.

II. BACKGROUND AND MATHEMATICAL PRELIMINARIES

A. HEI DECISION MAKING PROCESS

Higher Educational Institutes, autonomous in nature, are a kind of tertiary sector who are responsible for the governance and management of their finances, activities, and personnel. They autonomously own the responsibility of decisions about institutional priorities, strategies, goals, resource allocation, and accountability for these decisions [28]. Specifically, HEIs have three types of governance to manage the operational and managerial behaviors, including academic, bureaucratic, and corporate [29].

1) ACADEMIC

Faculty members hold the authority and decision making powers in academic activities like teaching, curriculum, academics, and administration.

2) BUREAUCRATIC

They have management layers with divisions of workforce characterized by procedures, fixed administration, and directly ordered by higher leaders.

3) CORPORATE

Students are considered the core customers of HEIs. The marketing activities required for retaining and acquiring students [30]. Decision making for marketing activities as required by other business enterprises to meet customer needs and market competition needed for HEIs.

B. COLOURED PETRI NETS

Petri nets originated from the dissertation by C. A Petri in 1962 [26]. It is a bipartite directed graph populated of places, transitions, and directed arcs. In Petri nets, places and transitions are connected with directed arcs. Petri nets provide an excellent modeling formalism for systems that are concurrent, asynchronous, distributed, parallel, or non-deterministic.

Definition 1: Formally Petri nets can be defined as a tuple $Pn = (Pl, Tr, In, Ot, M_0)$, where

(i) $Pl = \{pl1, pl2, \ldots, plm\}$ is a finite set of places;
(ii) $Tr = \{tr1, tr2, \ldots, trn\}$ is a finite set of transitions;
(iii) $In : Pl \times Tr \rightarrow Pn$ is an input function to denote directed arcs from places to transitions;
(iv) $Ot : Tr \times Pl \rightarrow Pn$ is an output function to denote directed arcs from transition to places; and;
(v) $M_0 : Pl \rightarrow Pn$ is the initial marking.

Coloured Petri Nets (CP-Nets) [25] are colored extension of standard Petri nets. In CP-Nets, colours are introduced to manage data types. CP-Nets have the expressive power of programming languages combined with standard Petri nets to leverage a powerful discrete event modeling formalism.

Definition 2: A Colored Petri Net is a tuple $N = (P, T, F, \Sigma, C, g, f, M_0)$, where:

(i) P is a finite set of places,
(ii) T is a finite set of non-empty transitions,
(iii) F is a finite set of arcs,
(iv) Σ is a finite set of non-empty colour sets,
(v) C is a colour function that assigns to each place. It is defined from P into Σ.
(vi) $g : T \rightarrow EXP$ is a guard function. It is an expression of Boolean type and assigned to each transition $t \in T$.
(vii) $f : F \rightarrow EXP$ is an arc expression function that is assigned to each arc $a \in F$
(viii) $M_0 : P \rightarrow EXP$ is an initialization function that assigns to each place $p \in P$.

Coloured Petri Nets are based on strong mathematical logic having roots in set theory, relations, and predicate calculus. They are useful in modeling both non-deterministic and stochastic processes. They construct a mathematical model and allow systematic exhaustive exploration of the model to analyze and prove the correctness of the system under consideration. On CP-Net, untimed models are mostly used to reveal logical errors while timed nets are used for concurrent systems.

Hierarchical Coloured Petri Nets (HCP-Net) are extensions of CP-Net to leverage the power of modular representation. It introduces the concept of hyper-transitions in the abstract model to hold in the Sub-models. The CP-Net comes with a user-friendly graphical interface. The CPN-tools [31] has a large community and is used in industrial projects as well as in research projects, has extensive documentation and support.

Definition 3: A hierarchical CP-Net is a finite set of nets $HCPN = S1, S2, S3, \ldots$. Each net S in $HCPN$ is a tuple $S = (P, T, F, \Sigma, C, g, f, M_0, h)$, where:

(i) F, Σ, C, g, f, M_0: are defined as in CP-Nets,
(ii) $P = ODP \cup INP$, where: ODP is a set of ordinary places as defined in CP-Nets, and INP is a set of interface places.
(iii) T $= ODT \cup HPT$, where: ODT is a set of ordinary transitions as defined in CP-nets, and HPT: a set of hyper-transitions.
(iv) G: is a **guard** function.
(v) h: is a function that maps each hyper-transition to a net.

The dynamic behavior of the CP-Nets are observed by firing the transitions. A transition is fired only when it is in the enabled mode. To enable a transition, some preconditions need to be satisfied. The preconditions are defined as label expressions on an arc or as a guard on transition. When a transition is fired, it updates the marking of the net. To understand the transition firing preconditions and marking operations, we need to discuss the following concepts.

We use a function $Var(Tran)$ to present set of variables used in guards for a transition *trans* or used in arcs expression labels on input or output arcs of *trans*.

Definition 4: Binding of a transition *tran* is defined as function *bn* on $Var(Tran)$, such that:

(i) For every $vertex \in Var(Tran) : bn(vertex) \in Type(vertex)$,
(ii) The binding of *Tran* satisfies the guard function of *Tran*.

Definition 5: A binding element is defined as a pair $(Tran, bn)$, such that *tran* is a transition and *bn* is a binding of *tran*.

Definition 6: Firing a hyper-transition *htran* defined in a net *HCPN* are the same as well as for an ordinary transition.

C. DECISION TREES

A decision tree, which is a mathematical decision support tool that represents the decisions in a tree structure. It builds a top-down model to classify instances of the given data set. A decision tree model trained on the data set is used to predict classification labels and categorize the objects present in the data set [32].

There is a root node in the decision tree at the top and leaf nodes at the bottom. The outgoing edges from the root, represent different possibilities and outcomes. The internal nodes represent the test attributes while the edges represent the result of the test. In the last stage, leaf nodes represent the predicted outcome of the internal node selection [33].

There are two stages in the decision tree construction process. The first stage of the process consists of decision tree training. In the second stage, the trained decision tree from the previous step is pruned. The decision tree building process is performed in top-down order. During the tree building process, the recursion method is used to partition the tree to put object instances of a similar nature to a single class label.

D. MODEL CHECKING

Model checking [34] is a formal verification technique based on formal methods for software and hardware verification.

The system, behavioral properties are modelled mathematically in the form of a state-transition system to guarantee the correctness of the system. A model of the system under consideration is developed using tools and techniques having a sound mathematical foundation for a comprehensive review. The model checking of the system requirement and design specifications ensure the correctness properties and increase the quality of the system. In the process of model checking, all execution paths and states are inspected automatically and exhaustively to ensure that there is no error state or deadlock-state.

E. DECISION TABLES

Decision Table [35] is a tool that supports solving complex computational problems. The decision tables represent problems composed of conditions, rules, and actions. In these tables, conditions are variables that need evaluation to perform specific actions. These can be a single step or a set of multiple situations that need to be chosen to depend upon the data in conditional variables.

III. LITERATURE REVIEW

A. MACHINE LEARNING IN EDUCATIONAL DATA ANALYSIS

Supervised Machine Learning algorithm applications in the educational field include prediction of grades, course selection, course recommendations, and students' drop-out chances. It has received tremendous attention from the research community [36]–[41]. The extraction of rules for decision-making based on machine learning was investigated in [42]–[44]. When a rule-based system makes errors, a key question is how to identify and correct the rules that cause these errors. An error can be defined as the discrepancy between the belief value generated by the system and that indicated by a presumed correct knowledge source [45]. Our proposed approach uses supervised machine learning, in particular decision trees, to produce decision rules and also applies the CP-Nets formalism to guarantee the correctness of decision rules.

Onan [15] proposed a recurrent neural network (RNN) based model for opinion mining on instructor evaluation reviews to aid the administrative decision-making process. The study indicated that deep learning-based architectures outperform the conventional machine learning classifiers for instructor reviews classification. Nieto *et al.* [1] discussed the management decision structure of higher educational institutes to support better governance using supervised machine learning classification algorithms. The decision tree, logistic regression, and random forest supervised applied classification algorithms to undergraduate engineering data sets. The results of the research were compared using the ROC curve to demonstrate outcomes. A novel model is proposed to address student's efforts and learnability from a generative perspective by exploiting temporal variations [46]. Francis and Babu [47] have proposed a hybrid predictive algorithm that uses both classification and clustering techniques to mine educational data sets to evaluate student performance in academia for higher educational institutes.

Miguéis *et al.* [48] proposed a two-stage predictive model for predicting students' career path at the end of the final year. We analysed the dataset of 2459 students from 2003 to 2015 to confirm that random forest outperforms other supervised classification algorithms. Jia *et al.* [49] proposed the SM-Naive-Bayes model to overcome the problem of low accuracy faced by classification techniques to predict student results. The developed model used previous stage course performance to predict future performance. Roy *et al.* [50] discussed student academics and courses to identify their interested career areas. The research motive is to identify primer career paths for computer science students.

B. CLASSIFICATION ERRORS

Intuitively, it is assumed that the classification labels are perfectly correct. However, in modern-day applications, it is observed that class labels have errors leading to a reduction in classification accuracy [20], [51].

In the literature, approaches have been discussed to mitigate classification errors caused by human mislabeling using genetic programming. In [52], binary classification is studied in the presence of erroneous labels. The authors proposed an approach to suitably modify any given surrogate loss function and the use of a simple weighted surrogate loss to obtain strong empirical risk bounds. Other applied theoretical works are presented in [53], [54], where the authors developed and analysed an improved logistic regression classifier that is robust to label noise.

Scott *et al.* [55] discussed conditions for consistent classification for data with label noise. In [21], weighting is discussed to train a classifier for noisy label data set, and [56] discussed loss factorization for learning with mislabelled data.

C. FORMAL VERIFICATION OF NEURAL NETWORKS

The formal verification of artificial neural networks (ANN), which has been explored extensively by the research community. In the literature, the majority of the work discussed the usage of theorem proving for formal verification of neural networks [57], [58]. Ivanov *et al.* [59] trained a nonlinear neural network for formal verification of the safety properties. They transformed a sigmoid based neural network into an equivalent hybrid system. Their proposed method works on many practical examples. In [60] Huang *et al.* proposed a formal verification methodology for the robustness of classification models. They used polyhedrons for capturing the intermediate perturbation present in neural network model.

In [57] Pulina and Tacchella used SMT solver for investigating the mis-classification in the output layer. They discussed several adversarial examples with the proposed methodology. Katz *et al.* [61] integrated the simplex method and SAT solver for formal verification of deep neural networks. They verified the safety properties of airborne

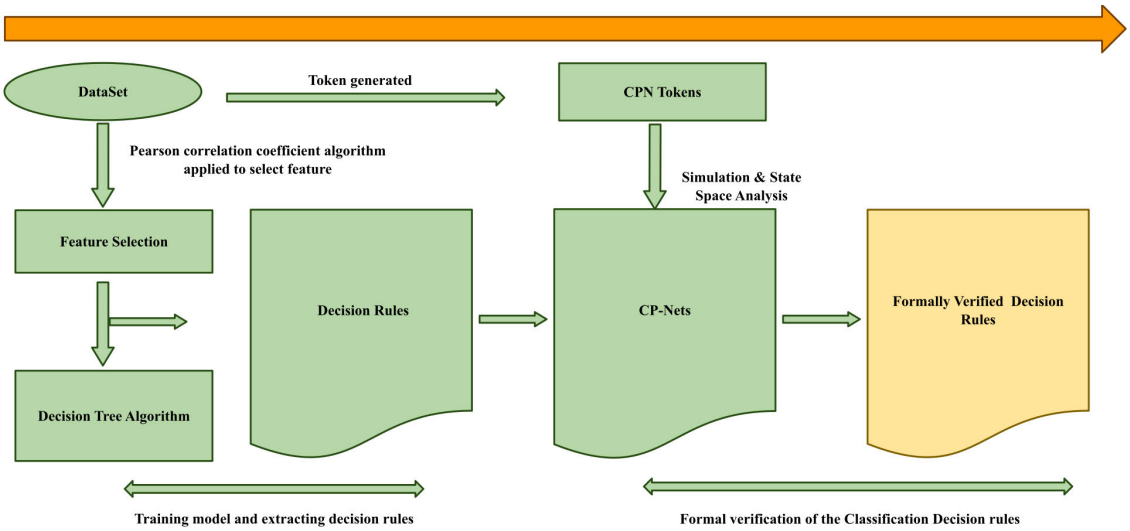


FIGURE 1. The proposed research methodology to formally verify decision rules using Coloured Petri-Nets.

collision avoidance systems. Ehlers [62] used an LP solver with a modified SAT solver used to verify neural network correctness. The method used a technique to approximate the network behavior to limit the search space. The methodology was implemented using two case studies having collision detection and digit recognition problems. In this research work, a similar verification problem is studied to formally verify the classification rules using CP-Nets as discussed above for neural networks.

D. FORMAL VERIFICATION OF DECISION TREES

There is little significant work present in the literature on decision tree formal verification. Törnblom and Nadjm-Tehrani [63] proposed a methodology to verify decision tree ensemble models' safety property. The input and output patterns were analysed using the VoTE tool by discussing two case studies. In [64] safety-critical properties of random forests are formally verified by partitioning the input space into disjoint sets for decision trees. The equivalence classes are computed from a random forest and verified against requirements.

In [65], Bastani *et al.* trained a neural network to play the game named Pong. An equivalent decision tree is extracted from the trained neural network as decision tree model verification is much easier than neural networks. The decision tree policies were formally verified by using an SMT solver. Our method uses CPN formalism to guarantee the correctness of decision rules extracted from the decision tree. The CP-Nets colour tokens simulate through model transition paths to verify decision rules' correctness to mitigate biasing and mislabelling errors.

IV. MATERIALS AND METHODS

The proposed approach used a supervised machine learning algorithm, a decision tree, to extract decision rules from the teaching data set. Secondly, it used CP-Nets formalism to ensure the correctness of decision rules as shown in Figure 1. In the first step, the decision tree is induced. Secondly, decision rules are extracted from generated trees, and, finally, CP-Nets formalism is applied to ensure the correctness of decision rules. Hence, the approach has three steps, first induce a decision tree, the second step is to extract decision rules from generated trees, and in the final stage, ensure the decision rule correctness and comprehensibility using CP-Nets formalism. The decision rule extraction is performed by traversing decision tree paths from the root node to leaf nodes. The CP-Nets model transformation from decision rules was carried out by applying Algorithm 1.

A. DATA COLLECTION

The data set consists of candidates' academic details for spring semester 2018 of a higher educational institute and available at the link[1]. The data set contains academic details of applicants seeking admission in six departments. The data is exported in comma-separated values (CSV) format from the admission system of the institute. The data set contained 4621 records with 108 features. The data set contains 555 instances labeled as **"Yes"** and 4066 labeled as **"No"**. The goal is to find decision rules that classify an applicant's admission status as a **"Yes"** or **"No"** label. The label **"Yes"** represents labels for students offered admission in a particular department; while the label **"No"** shows that the applicant was not offered admission. These decision rules are exploited to monitor and evaluate the admission process for specific departments. These decision rules leverage the HEIs decision-makers to analyse the applicant admission process effectively. It leverages improved decision-making powers to academic administration for maximizing student intakes

[1]https://github.com/mhmmd-nauman/DataSets/raw/master/spring18/dataset/department_wise_csv_data.rar

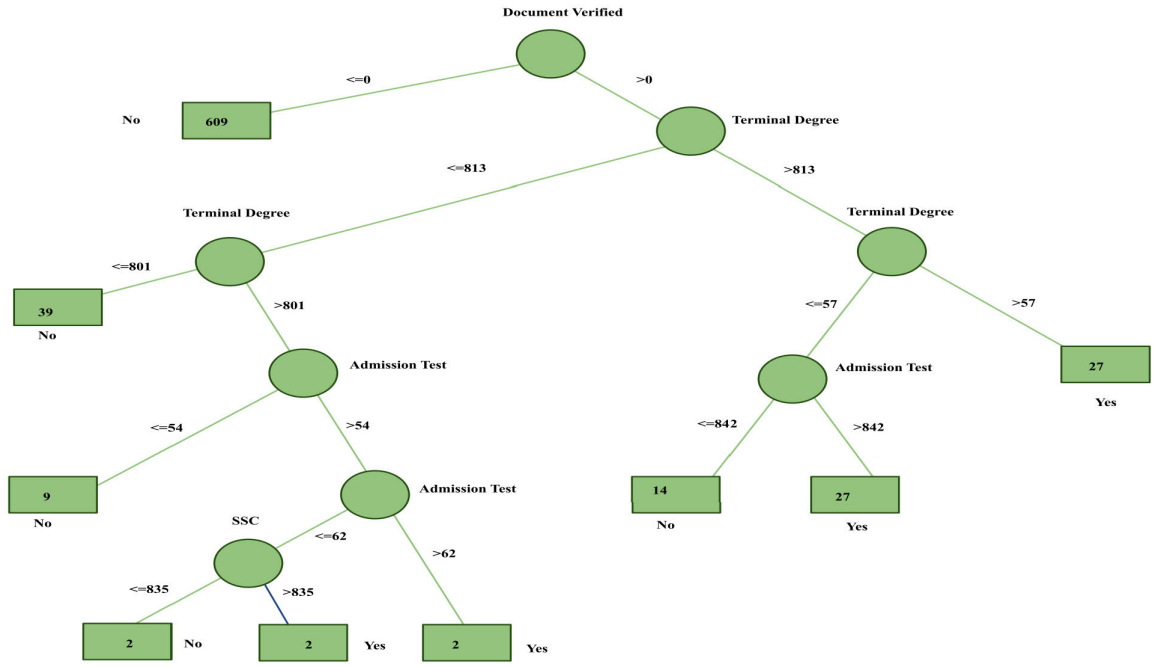


FIGURE 2. Decision tree showing the binary classification results for the Computer Science department.

TABLE 1. Data set selected features using Pearson correlation coefficient.

Feature Name	Feature Description	Type
Application Status	It represents the status of application.	Dichotomous
Secondary School Certificate	Secondary School Certificate (SSC) represents the candidate 10th grade score.	Continuous
Admission Test	Admission Test (AT) represents the grades obtained by candidates in the admission test conducted.	Continuous
Terminal Degree	Terminal Degree (TD) represents the grades obtained by candidates in the relevant degree required for admission in an offered program.	Continuous
Document Verified	Document Verified (DV) represents the weather the candidate is appeared in document verification process.	Dichotomous

and maximizing the use of available resources in a specific department. The decision tree is induced to extract decision rules. The decision tree is chosen because of the organized structure and better rule visualization for data split points. The decision tree in Fig. 2 trained on the academic data set of the candidate in the computer science department. It should be noted that the decision tree model consists of nine leaf nodes that can be considered decision rules.

B. FEATURE SELECTION

Finding the most correlated attributes to the final class label and how much they affect the classification is crucial. Significantly, this stage shows the average correlation of the prominent features of the predicted class. Therefore, to extract more comprehensible rules, irrelevant features need to be filtered out. Data fields with high correlation were considered as recommendation points for students and academic staff. The Pearson correlation coefficient algorithm was used to select the related features.

After comparing the selected features with each other for correlation, the top 5 high related features were chosen including Document Verified, Terminal Degree, Admission Test, SSC. For two quantitative features X and Y, the correlation is measured using Pearson's correlation coefficient, which is defined as

$$r_{xy} = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{(n-1)S_x S_y} \tag{1}$$

where x and y are sample means for X and Y respectively, S_x and S_y are sample standard deviations for X and Y, and n is the size of the sample used to compute the correlation coefficient. Table 1 shows the highly correlated features selected using the Pearson correlation coefficient algorithm. Table 2 shows the sample rows from the data set used to train the tree model.

C. DECISION TREE MODEL

J48 algorithm is applied to induce the classification models with a ten-fold cross-validation procedure. J48 is an enhancement to the ID3 algorithm [66] and open-source implementation of C4.5 algorithm [67]. C4.5 uses the gain ratio criterion, which is based on information theory and produces suboptimal trees heuristically.

TABLE 2. Sample data set used for training decision tree classification models for Computer Science department.

Sr	Status	SSC	AT	TD	DV	Label
1	No	637	0	667	0	No
2	Complete	1050	59	1100	1	Yes
3	No	916	60	825	0	No
4	Complete	803	59	783	0	No
5	No	813	44	1050	0	No
6	Complete	745	42	614	0	No

The WEKA [68] graphical interface is used to visualize decision trees. It is selected because it has a graphical interface and is very easy to use, and convenient to load and process data sets. It's easy to apply classification algorithms to train a predictive model and visualize it.

It also offers stability among precision, speed, and result interpretation. Table 3 shows a set of decision rules extracted from induced decision trees. The decision rules are extracted using path traversing from the root node to leaf nodes. In the decision, tree each leaf node represents a decision rule. The usefulness of a decision rule is usually summarized in two factors: support and accuracy. The support of a decision rule is the number of instances that satisfy a rule. The accuracy of a rule is a measure of how accurate the rule is in predicting the correct class for the instances to which the condition of the rule applies. The threshold value for support to consider a leaf node for a valid decision rule was set to five intuitively which reduced the number of decision rules to six for nine leaf nodes. The decision tree in Fig. 2 exhibits decision points for the computer science department data set in the tree structure.

D. CP-NETS MODEL FOR DECISION RULES

Petri-nets are used to model a system with discrete events, concurrency, and non-determinism. Hence, the first step towards modelling the system under consideration is to observe a set of events that cause a change in the system state. Therefore, to construct a CP-Nets model, decision choices in a system needed to be identified. These decision choices split the process into branches. The split choices are called decision points and define the basis for the decision rules in the system. Based on the data set under consideration, our primary goal is to guarantee the correctness of decision points for **"Yes"** or **"No"** labels. Therefore, CP-Nets are constructed for modelling decision rules. In the Petri Nets model, decision rules are modelled by transitions. The rule conditions are modelled by some input places, some expressions on input arcs, and some associated guards. All CPN models are accessible on the[2].

E. FORMAL VERIFICATION OF DECISION RULES

Decision rules are transformed into an equivalent CP-Net model for proving formal correctness guarantees by implementing algorithm 1. The CP-Net model consists of transition places and a set of colour tokens constructed using the extracted rules from the decision tree model. The rule conditions are transformed as guards on rule transitions (refer to algorithm 1).

Algorithm 1 Decision Rules to CP-Nets

```
Input: Rule Set and Class Label
Output: CP-Net Submodule
for decision rule in Rule Set do
    Create CP-Net Transition
    for rule.attributes in rule do
        Add transition guard for rule.attribute
    end
end
```

The abstract Hierarchical CP-Nets model of the proposed approach is illustrated in Fig. 3. The Hierarchical CP-Net model consists of three sub-models, namely *Classifier_Yes_Label*, *Classifier_No_Label*, and *Merge_Actual_Predicted_Labels*. The interaction between these sub modules is modelled through interface places. This model gives us an overview of the decision rules and the formal verification process, where all events are defined. It should be noted that in Fig. 3 the transitions *Classifier_Yes_Label*, *Classifier_No_Label*, and *Merge_Actual_Predicted_Labels* are hyper-transitions. Each transition represents one of the sub-model. The decision rule correctness is proven by simulating the colour token through the equivalent CP-Net model by implementing algorithm 2.

In sub-modules, decision rules are grouped hierarchically for better understanding and model visualization. Fig. 4 (a) shows **"Yes"** decision rules CP-Net sub-module before simulating colour tokens. The correctness of sub-modules is proved by simulating colour tokens through the CP-Nets model. The paths taken by colour tokens are observed and exploited to correct labeling and bias errors present in the data as discussed in the algorithm 2. The simulation of color tokens depends on the sequence of decisions within the model. Fig. 4 (b) illustrates the corrected model. Fig. 5 shows **"No"** label decision rules before and after simulating colour tokens, respectively. Notably, Fig. 5 (b) illustrates CP-Net model after simulation with correctness implementation using Algorithm 2. Intuitively, the decision rule correctness algorithm can modify, add, or remove the CP-Net model rules under consideration.

The color tokens created to represent the candidate's academic details in the CP-Nets are presented in table 4. The table 5 shows the definitions of the color sets used to model the decision rules into a CP-Net.

F. CORRECTNESS ALGORITHM DESCRIPTION

The correctness algorithm starts with input and output descriptions. The algorithm's input consists of CP-Nets and colour tokens. The variable *CP-Nets* represents the derived decision rules in terms of CP-Nets. The variable *CPN token*

[2]https://github.com/mhmmd-nauman/DataSets/raw/master/spring18/dataset/cp-nets-models.rar

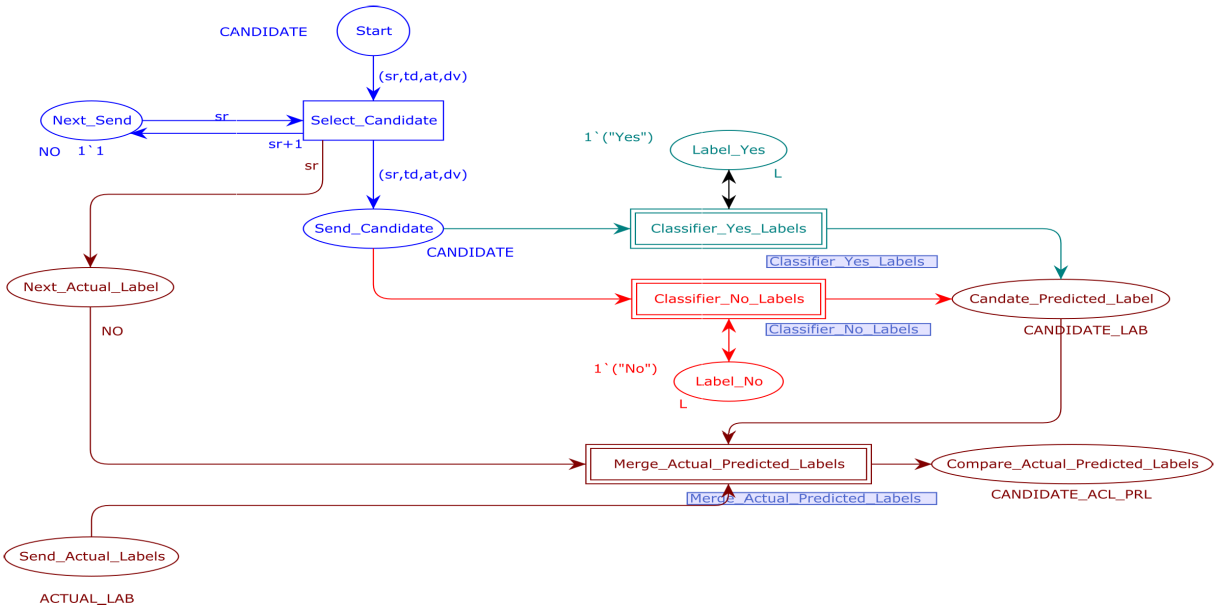


FIGURE 3. Hierarchical Coloured Petri nets abstract model for decision rule verification.

TABLE 3. Decision table showing the classification rules extracted from the trained decision tree model (shown in Fig. 2) for computer science department.

No	Label	Extracted rules	Rule nodes	Rule support	Rule accuracy
R1	Yes	Document-Verification >0 AND Terminal-Degree >813 AND Admission-Test >57	3	27	100
R2	Yes	Document-Verification >0 AND Terminal-Degree >842 AND Admission-Test <57	3	24	91
R3	No	Document-Verification >0 AND Terminal-Degree <801	2	39	100
R4	Yes	Document-Verification >0 AND Terminal-Degree $\leq$ 842 AND Admission-Test $\leq$ 57	3	49	10
R5	No	Document-Verification $\leq$ 0 AND Terminal-Degree $\leq$ 813 AND Admission-Test $\leq$ 54	3	395	100
R6	No	Document-Verification $\leq$ 0	1	609	100

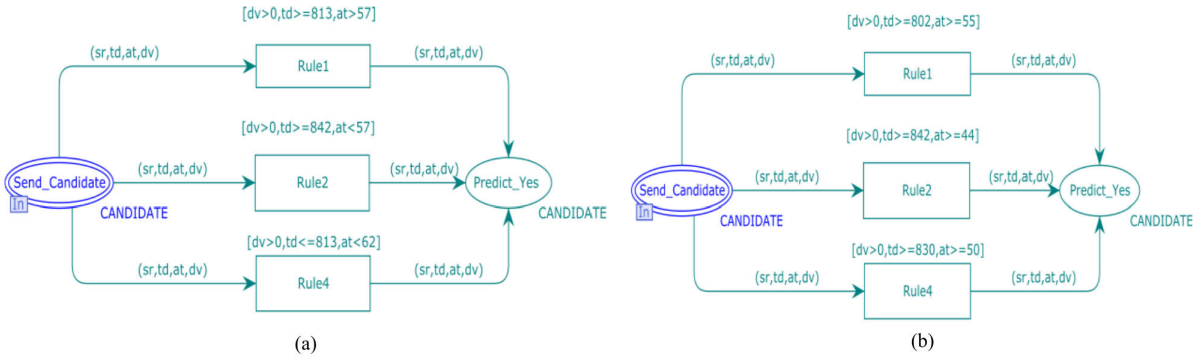


FIGURE 4. Hierarchical CP-Nets model for decision rules for class label "Yes" (a) Before formal verification rules (b) After formal verification rules.

contains the structure of the colour tokens used for formally verifying the CP-Nets model. This algorithm includes two loops that iterate through to simulate all tokens through all 6 rule transitions present in the CP-Net model (see lines 1-17). The transitions in the CP-Nets model abstractly decision rules. In the loop body, the first *IF Else* statement checks if a token is simulated through transition states for classification labels **"Yes"** or **"No"** in the model. First *IF* statement checks if the actual and predicted labels are comparatively the same, i.e. **"Yes"** token simulates to a transition state **"Yes"**,

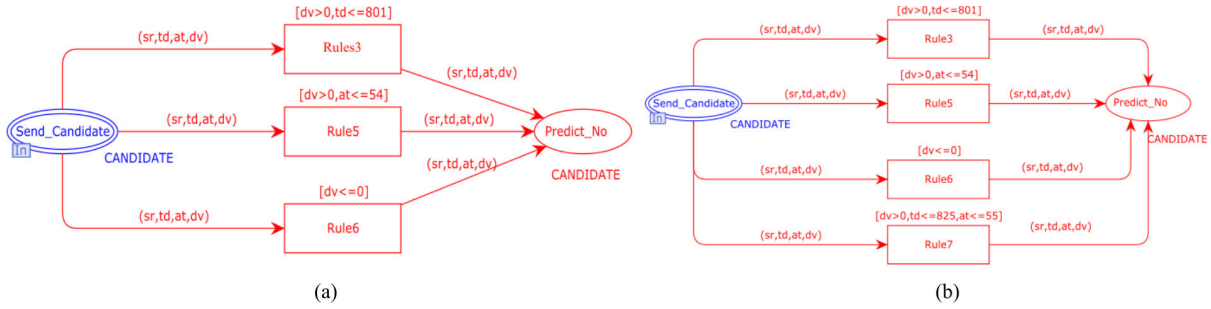


FIGURE 5. Hierarchical CP-Nets model for decision rules for class label "No" (a) CP-Nets model of decision rules before formal verification (b) CP-Nets model of decision rules after formal verification.

TABLE 4. Computer science department colour tokens classification labels before and after simulations.

		DT	CP-Nets	
Colour Token	Label	Predictive Label		Remarks
1‘(1,667,0,0)++	No	No	No	Verified
1‘(2,1100,59,1)++	Yes	Yes	Yes	Verified
1‘(3,825,60,0)++	No	No	No	Verified
1‘(10,841,49,1)++	No	Mislabel	Mislabel	Not verified
1‘(24,893,54,1)++	Yes	Yes	Yes	Verified
1‘(47,829,56,1)++	No	Mislabel	No	Corrected
1‘(65,815,60,1)++	Yes	Yes	Yes	Verified
1‘(75,699,60,0)++	No	No	No	Verified

marking the decision rule as correct. If the token actual and predicted labels are comparatively not the same, then the decision rule needs revision (see line 7). In that case, the routine *ReviseRule* is called to perform revision operations on equivalent transition for modification of CP-Net model.

The *ReviseRule* subroutine receives a *ruleset* variable and a colour token is simulated for correction verification. The colour token is again simulated through a model with revised rules to validate the actual and predicted labels. The rule set contains all rules in the CP-Nets model, and the token contains academic details. All rules are revised sequentially to simulate the token through the model to reach a transition state with a desired predictive token label (see lines 18-32). The *ReviseRule* subroutine contains two loops to mathematically adjust the classification decision rules attributes in the CP-Nets model. The *ReviseRule* subroutine performs mathematical operations to adjust rule attributes by incrementing or decrementing a value by 10 (see lines 22 and 24). The integer value 10 is chosen because the rule revising routine should not go out of bounds for other tokens.

In the case of *ReviseRule* subroutine call, a new version of the CP-Nets model is constructed to pass all colour tokens from the model to execute the formal verification algorithm again. This process continues until all rules are adjusted to pass the maximum number of tokens from the CP-Nets model. The final version of the CP-Nets model represents formally correct classification decision rules.

Algorithm 2 Decision Rule Correctness Process

Input: CP-Nets Model and Colour Tokens
Output: Formally Verified CP-Nets Model

```
1  for Token in Token Set do
2      for Rules in CP-Nets do
3          if States.Yes ← Token then
4              if Token.label = Yes then
5                  Rules ← Verified
6              else
7                  Rules ← ReviseRule(Rules,Token)
8              end
9          else
10             if Token.label = No then
11                 Rule ← Verified
12             else
13                 Rule ← ReviseRule(Rules,Token)
14             end
15         end
16     end
17 end
18 Function ReviseRule(RuleSet,Token)
19 for Rule in RuleSet do
20     for Attributes in Rule do
21         if Rule.attributes > Token.attribute then
22             Rule.attributes ← Rule.attributes − 10
23         else
24             Rule.attributes ← Rule.attributes + 10
25         end
26     end
27     if Token is simulated successfully then
28         return true
29     else
30         Add new rule into the RuleSet
31     end
32 end
```

G. DECISION RULES CORRECTNESS VERIFICATION

CPN tools provide a simulation mechanism to walk through Petri net models' reachability graphs to verify the model formally. A strong point of simulation is its flexibility as an analysis technique. Simulation is helpful in almost every situation. We assume that such a walk is representative of the behaviour of the modelled system. The disadvantages of simulation are that it can be time-consuming; sometimes,

TABLE 5. Definition of color sets for CP-Nets models shown in Fig.3.

Color Set Definition	Description
colset NO = INT;	Specifies the serial number for candidate.
colset DV = INT;	Specifies the document verification status of the candidate.
colset TD = INT;	Specifies the terminal degree grades obtained by the candidate.
colset AT = INT;	Specifies the admission test marks obtained by the candidate.
colset DOMICILE = string;	Specifies the applicant domicile certificate.
colset L = string;	Specifies the applicant classification label.
colset CANDIDATE = product NO*TD *AT*DV*L;	Cartesian product specifies the candidate attributes present in selected feature list.
colset CANDIDATE_LAB = product NO*TD *AT*DV*L;	Cartesian product specifies the candidate with class label attribute.
colset CANDIDATE_ACL_PRL = product NO*TD *AT*DV*L*L;	Cartesian product specifies the candidate with actual and predicted class label attributes.
colset ACTUAL_LAB = product NO*L;	Cartesian product specifies the actual label for a candidate.
var ac_sr,sr : NO;	Variable declarations for serial number.
var td : TD ;	Variable declarations for terminal degree grades.
var at : AT;	Variable declarations for admission test grades.
var dv : DV;	Variable declarations for document verification status.
var ac_label,label : L;	Variable declarations for actual and predicted labels.

long simulation runs are necessary to get reliable results. The simulation does not provide formal proof; that is, the simulation can only analyse an error's presence but never its absence. The absence of error in the model is verified using state-space analysis.

The model was simulated using a simulation tool available in the CPN tool graphical user interface. The simulation tools can simulate 50 colour tokens at a time. The transition places represent the applicant's status in the CP-Net model. The model simulation begins at the first transition state called initial marking. The single transitions in the model represent a single decision rule. The formal verification using simulation is performed in three steps. First, the model is loaded for execution with all colour tokens at initial marking. Second, colour tokens simulate through transition places. The transition places pass only colour tokens that verify the rule condition in the CP-Net model. In the third and final step, colour tokens are analysed that are simulated to *Compare_Actual_Predicted_Labels* places representing the applicant's predicted and actual labels. The sub-module *Merge_Actual_Predicted_Labels* shown in Fig. 6 (b) is used to analyse and compare the actual and predicted labels. The transition place *Compare_Actual_Predicted_Labels* in the model is reached by satisfying the classification decision rule.

The correctness of the classification decision rule is verified by simulating all colour tokens having the applicant's academic details. Table 6 shows revised comprehensible decision rules with remarks after performing simulations. The remarks column shows the rule revision status in the model. In Fig. 4 (b) & Fig. 5 (b), the transition place represents the revised rules obtained from the simulation process.

H. STATE APACE ANALYSIS

A state space, which represents all possible executions in the system under consideration, can be used for formally verifying the correctness guarantees and proving the absence of error mathematically. It is used to explore the dynamic properties of CP-Nets and formally verify a specific system property. Therefore, a state space is a directed graph representing each reachable marking and an arc for each occurring binding element.

In CPN Tools, standard state-space reports facilitate initial information about the properties of the model analysis. The state-space report includes statistics, boundedness property, home property, liveness property, and the CP-Net model's fairness properties.

The statistical information on a state-space shows the number of nodes and arcs of state space, construction time, and Scc graph. The boundedness property specifies how many and which tokens a place may hold when all reachable markings are considered. The home property shows us home markings. A home marking is a marking that can be reached from any reachable marking.

The liveness property provides information about dead markings, dead transitions, and live transition. In dead markings, there is no transition enabled. A transition is dead if there are no reachable markings from that transition. There is always an occurrence of a sequence containing a transition from any reachable marking in live transitions.

The fairness property provides information about transitions with infinite occurrence sequences. This part of the state space report lists those transitions that have infinite occurrence sequences.

Fig. 6 (a) shows a partial section of the entire model's state space report with 50 tokens from the Department of Computer Science data set. The analysis report has 21620 markings and 92565 arcs. This report is generated automatically by a state space tool in 300 seconds. The exact number of markings and arcs in the Scc graph shows the absence of cycles in the state space. There are no home markings under the home properties section. The liveness properties section has a list of 6 dead markings, a single dead transition, and no live transitions. Notably, there are no infinite occurrence sequences as defined by the fairness properties in the report. Accordingly, home properties, number of live transitions, and fairness property are desirable. The reachability graph computed by the CPN - tool is analysed for formal verification of specific properties and discussed in the below section.

1) LABEL **"YES"** VERIFICATION SUB-MODULE

Property 1 ("Yes" Classifier Always Predicts the Correct Label): The property states that if a token meets the

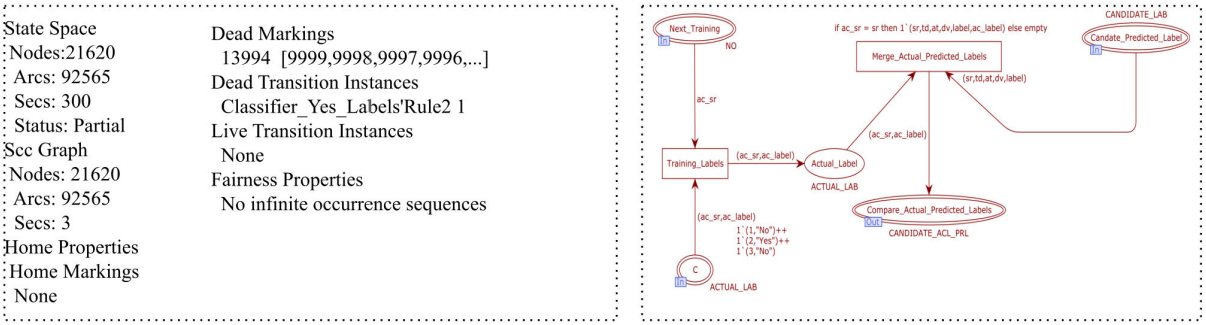


FIGURE 6. Part of the state space report in (a) with CP-Nets model for Actual and Predicted labels module in (b).

TABLE 6. Decision Table showing the revised decision rules after performing formal verification of decision rules in the computer science department.

No	Label	Extracted rules	Remarks	Rule nodes	Rule support	Rule accuracy
R1	Yes	Document-Verification >0 AND Terminal-Degree $\geq$ 802 AND Admission-Test $\geq$ 55	Rule Revised	3	44	89
R2	Yes	Document-Verification >0 AND Terminal-Degree $\geq$ 842 AND Admission-Test $\geq$ 44	Rule Revised	3	45	96
R3	No	Document-Verification >0 AND Terminal-Degree $\leq$ 801	No Changes	2	39	100
R4	Yes	Document-Verification >0 AND Terminal-Degree $\geq$ 830 AND Admission-Test $\geq$ 50	Rule Revised	3	46	93
R5	No	Document-Verification >0 AND Admission-Test $\leq$ 54	No Changes	2	2	100
R6	No	Document-Verification $\leq$ 0	No Changes	1	609	100
R7	No	Document-Verification >0 AND Terminal-Degree $\leq$ 825 AND Admission-Test $\leq$ 55	New Rule	3	36	97

conditions for the **"Yes"** class, then the model should put a predicted label **"Yes"** on it and send it to *Candidate_Predicted_Label* place. The marking of the place *Candidate_Predicted_Label* proves that all colour tokens are labelled as **"Yes"** that flows through transitions having guards as rule logic in *Classifier_Yes_Labels* sub-modules (R1, R2, R4).

Property 2 (Classifiers Put "Yes" Labels on Colour Tokens for Labeling Them According to Decision Rules): The marking of the place *Label_Yes* on transition *Label_Cand_Yes* proves that colour tokens are always labelled as **"Yes"** on the place *Candidate_Predicted_Label*.

2) LABEL **"NO"** VERIFICATION SUB-MODULE

Property 1 ("No" Classifier Always Predicts Correct Labels): The property states that if a token meets the conditions for the **"No"** class, then the model should put a predicted label **"No"** on it and send it to *Candidate_Predicted_Label* place. The marking of place *Candidate_Predicted_Label* proves that all colour tokens are labelled as **"No"** that flows through transitions having guards as rule logic in *Classifier_No_Labels* sub-modules (R3, R5, R6).

Property 2 (Classifiers Put "No" Label on Colour Tokens for Labeling Them According to Classification Rules): The marking of the place *Label_No* on transition *Label_Cand_No* proves that colour tokens are always labeled as **"No"** with the place *Candidate_Predicted_Label*.

TABLE 7. Confusion matrix.

		Predicted Labels	
		No	Yes
Actual Labels	No	TP	TN
	Yes	FP	FN

3) ACTUAL AND PREDICTED LABELS VERIFICATION SUB-MODULE

Property 1 (The Actual and Predicted Label Token Reaches Compare_Actual_Predicted_Labels *Place):* The property states that colour tokens after passing through classification rule transitions should reach *Compare_Actual_Predicted_Labels* place to comparison of actual and predicted labels. It should be noted that at this place the labels are compared for guaranteeing the correctness of decision rules.

The marking of place *Compare_Actual_Predicted_Labels* place proves that a token with actual and predicted labels always reaches the specified place.

I. PERFORMANCE METRICS

Decision tree classification model utilized to label applicants either as **"Yes"** and **"No"**. It should be noted that the classification label **"Yes"** represents the candidate offered admission in a particular department, and **"No"** shows an applicant not offered admission. For classifying instances, the confusion matrix was calculated as shown in Table 7.

TABLE 8. Decision tree classification models performance analysis before and after performing formal verification.

	Decision Trees				CP-Nets formalism			
Department	TNR	TPR	ACC	AUC	TNR	TPR	ACC	AUC
Computer Science	86.44	98.36	97.40	0.92	89.39	100	99.02	0.94
Agriculture	98.03	93.00	94.95	0.95	91.30	99.46	96.01	0.95
Biochemistry & Biotechnology	98.38	99.63	99.54	0.99	98.36	99.87	99.77	0.99
Chemistry	83.01	99.01	98.13	0.91	88.67	99.88	99.26	0.94
Physics	92.42	99.29	98.58	0.95	100	97.36	97.58	0.98
English	91.80	99.12	98.52	0.95	96.66	99.85	99.59	0.98
Complete Data set	94.05	98.50	97.96	0.96	92.94	99.5	98.68	0.96

For evaluating and comparing the classification model's precision before and after applying CP-Nets formalism, we have chosen different parameters. Including the number of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN), Accuracy (ACC), Precision (P), Recall (R), Sensitivity (TPR), Specificity (TNR) and Area under ROC curve (AUC) as designated quantitative metric. Here the total number of instances is represented as N and computed such as:

$$N = TP + TN + FP + FN \tag{2}$$

Accuracy (ACC): ACC specifies the accuracy of the classification model such that:

$$ACC = (TP + TN)/N \tag{3}$$

Precision (P): P specifies the number of positive instances correctly identified from all positively predicted instances such as:

$$P = TP/(TP + FP) \tag{4}$$

Recall(R): R specifies the number of positive instances identified from all positive instances such as:

$$R = TP/(TP + FN) \tag{5}$$

Sensitivity (TPR): TPR specifies the correct classification rate of positive instances
such as:

$$TNR = TP/(TP + FN) \tag{6}$$

Specificity (TNR): TNR specifies the correct classification rate of negative instances
such as:

$$TNR = TN/(TN + FP) \tag{7}$$

Area under the ROC curve (AUC): ROC curve graphically displays the performance of the predictive model by plotting the true positive and false negative rates for all possible values. The area under the curve shows the overall performance of a classification model. The perfect prediction value for AUC is 1.0 and 0.5 corresponds to the diagonal on the ROC curve.

TABLE 9. Comparisons with other supervised machine learning methods on computer science department data set.

Method	TPR	TNR	ACC
ANN	77.14	100	97.81
SVM	80.95	99.49	97.86
K-NN	44.44	94.64	90.90
RF	81.52	99.21	97.63
NB	100	87.93	89.09
Proposed approach	100	89.39	99.02

TABLE 10. Comparisons with other rule extraction methods on computer science department data set.

Method	Rules	TPR	TNR	ACC
Decision Tables	26	98.51	84.74	97.40
OneR	1	97.17	33.89	92.06
PART	5	98.80	84.74	97.67
RIPPER	3	98.95	89.83	98.22
Proposed approach	7	100	89.39	99.02

J. WHAT-IF ANALYSIS

What-if-analysis is used to structure the scenario case as "what will happen to the solution if an input value is changed?" As discussed in [69]. In the current data set analysis, the prominent variables for what-if-analysis are shown in Table 1. It is observed from the first decision rule in Table 6 that applicants who have marks greater than 802 out of 1100 in the terminal degree with an admission test mark greater than 55 are classified to the label **"Yes"** for the computer science department. The second decision rule states that an applicant needed 842 marks in the terminal degree with admission test marks greater than 44 out of 100. The third rule states that if an applicant has less than 801 marks they are not offered admission. The fourth rule states that if they have scored greater than 50 marks in the admission test, then they need greater than 830 marks in the terminal degree to have a strong chance to be selected by the computer science department. It is observed from rule 6 that document verification is a very crucial part of the admission process and if any applicant has not verified their original academic documents, they are not offered admission in the department. Rule 5 states that if an applicant has less than 54 marks in the admission test, then they are not offered admission. It is evident from Rule 7 that if an applicant has less than

825 marks in the terminal degree and less than 55 in the admission test, then there is no chance of selection in the computer science department.

K. RESULTS

The data set contains the applicant's details from six departments of a higher educational institute. The decision tree is induced using the J48 algorithm and shown in Fig. 2. The decision tree is visualized using WEKA Tool; the decision rules were extracted by traversing edges of the decision tree and shown in Table 3. The threshold value for support to consider a leaf node for a valid decision rule is set to five intuitively to reduce the number of decision rules for consideration. These decision rules represented the scenarios of admission selection criteria for applicants in the respective department. The derived rules are transformed into CP-Nets transition rules by using algorithm 1. The Hierarchical CP-Nets model of the proposed approach is shown in Fig. 3. The colour tokens are constructed using the same data set as used to induce decision tree models. The remarks column shows whether a token simulated through a decision rule provides a correctness guarantee or identifies the rule as incorrect or biased. The comparative CP-Nets models before and after colour token simulation analyses for comprehensible classification decision rule analyses are shown in Fig. 4 and Fig. 5.

In Table 3, decision rule No. 1, No. 2 and No. 4 with a **"Yes"** label, can be exploited to predict the applicant selection for admission in a particular department. The classification accuracy for rules No. 1 and No. 2 achieves more than 91%, but No. 4 has an accuracy of 10% only, which shows the presence of mislabelled or biased errors in the data. Hence, to mitigate such error, this work proposed use of the CP-Net model simulations. Accordingly, rules No. 3, No. 5 and No. 6 with a **"No"** label can be employed to predict the applicant for not selection cases. The classification accuracy of these rules is 100%.

Algorithm 2 is used for comprehensible classification decision rule analyses and verification. In Table 6, decision rules No. 1, No. 2, and No. 4 with label **"Yes"** are verified using the simulation process for guaranteeing the correctness of decision points. The rules have the rule accuracy of nearly 90%, and they cover most of the instances with **"Yes"** label. We should note that the application of simulation processes has corrected the decision rules and enhanced their generality by adjusting the rules, which led to increased rules accuracy and a large number of data sent instances passed through the models. Rules No.3, No.5, No.6, and No.7 are with a **"No"** label to predict admission not offered case. The rule sets accuracy for the **"No"** label is 99%.

Notably, the process of academic document verification documents is a critical bottleneck in the admission process because 609 candidates cannot complete this process in the computer science department only. Hence, the institute needs a revised policy for academic document verification. Accordingly, a set of understandable decision rules will assist HEIs administrative authorities in effective and efficient academic decision making. Therefore, this research work findings help HEIs educational administration monitor, improve, and redesign learning activities. Table 8 shows the performance evaluation results obtained by applying the proposed approach on six-department applicant's data.

According to the results in Table 9, it is observed that the proposed approach outperforms Artificial neural networks, Support vector machine, K-Nearest neighbours, Random forest, and Naive Bayes in terms of accuracy, sensitivity and area under the curve. It has performed marginally low in terms of specificity for baseline classifiers. Therefore, the proposed approach shows the primary advantages of good classification ability on the data set.

From Table 10, we can see that the proposed approach shows the best performance in terms of sensitivity, specificity, accuracy, and area under the curve against the other rule extraction methods.

We can observe from Table 11 that the proposed approach has guaranteed the correctness of decision rules by mitigating the bias effect. Fig. 7 (a) & (b) illustrates the confusion matrix performance evaluation before and after the formal verification process. It is noticeable from the confusion matrix that the number of correct classified instances has increased by applying the CP-Net simulations on the decision tree equivalent model. Fig. 7 (c) & (d) shows the ROC curve to evaluate the performance of the proposed approach for all departments under consideration. The ROC curve in green represents the performance evaluations before formal verification, and the red color curve represents the performance evaluation after formal verification performed on decision rules.

L. DISCUSSION

Even though the use of Machine Learning is still emerging in the educational field, it can effectively be applied to analyse the information. It can leverage HEIs directors with improved understanding to make decision making by providing predictions, classification, visualization, and decision rules. Our proposed approach accuracy for academic data analyses is above 98% as shown in Table 6. Table 11 shows the correctness improvements gained using CP-Net simulations. This research work concludes that supervised machine learning classification models are a good choice for educational data analysis.

Our approach outperforms the decision tree model proposed in [1]. In [1] strategic decision making has been focused using three supervised machine learning algorithms, including decision tree, logistic regression, and random forest with 84% overall accuracy. The object of the research was to predict the graduation rates of students.

We have applied the decision tree algorithm with CP-Net formalism to obtain decision rules the accuracy over 98%. In [10] association between internet usage behaviour and academic performance has been predicted using machine learning. The study results indicated that behaviour discipline plays a vital role in academic success. The proposed approach used a decision tree, neural network, and support

TABLE 11. Decision rule correctness analysis.

	Decision Trees			CP-Nets formalism		
Rule No	nodes	Covered instances	Accuracy	nodes	Covered instances	Accuracy
1	3	27	100	3	44	89
2	3	24	91	3	45	96
3	2	39	100	2	39	100
4	3	49	10	3	46	93
5	3	395	100	2	2	100
6	1	609	100	1	609	100
7	-	-	-	3	36	97
Average	2.5	190.5	83	2.42	117.26	96

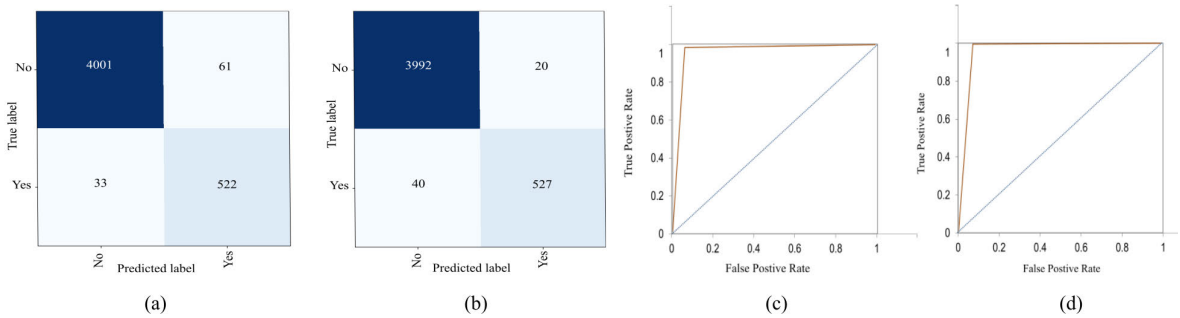


FIGURE 7. Confusion Matrix of classification model before and after formal verification.

vector machine algorithm to predict a student's study performance. In [47], a prediction algorithm based on classification and clustering techniques for evaluating a student's academic performance has been discussed. The research proposed a mixed-use decision tree, SVM, neural networks and k-mean clustering algorithms for performance evaluation. In [70] higher educational institutes' academic decision variables are discussed. The use of supervised machine learning algorithms, including neural networks and support vector machines, is explored for predicting graduation rates for decision making. We used a decision tree with CP-Net formalism to evaluate the academic background of students for decision making.

In the literature, correctness guarantees for machine learning outcomes are discussed in [63], [71], [72]. In [63], a methodology to extract equivalence classes from decision trees and tree ensembles to formally verify input-output mappings has been discussed. The authors presented the tool VoTE (Verifier of Tree Ensembles) method and evaluated it on two case studies. In [72], the authors proposed re-weighting the data points without changing the labels to correct label biases present in the data. Our work used CP-Net model simulations and state-space analysis to verify the correctness of classification rules formally. It also mitigates classification biasing and mislabelling errors penetrates in decision rules.

In formal verification techniques, model checking and theorem proving are prominent to verify supervised machine learning algorithms. This research work is the first attempt to guarantee correctness with CP-Net model simulations to the best of our knowledge. Empirically, it is evident that formal verification with CP-Net models improves the accuracy of classification outcomes. The accuracy improvements are gained by mitigating mislabelling, and data bias errors. Hence, the supervised machine learning model implemented in the educational decision support system improves decision-making quality. When used for administrative decision support, our model can provide a simulation tool to explore various scenarios and identify the rules and bottlenecks in the admission process.

It is essential to state that this study's results may not generalize to a larger scale as institutional bias may be affected. Therefore, further work comparing results with extended data sets from other institutes would be beneficial. Despite this limitation, achieved outcomes remain significant in the area of educational data analysis.

The results of this research study provide insights for enhancing and revising the administration's decision making. To our best knowledge, this study is the only work aimed at training decision tree models and guaranteeing the correctness of comprehensible decision rules using formal verification.

V. CONCLUSION

Supervised Machine Learning, specifically decision trees, to analyse educational data can support higher educational institutes' better decision-making. It can facilitate in unveiling the decision rules present in large amounts of educational data stored at higher educational institutes. The organized structure and better rule visualization put the decision tree

algorithm in one of the best choices for analysing educational data sets. However, machine learning outcomes have been prone to bias and mislabelling errors to mitigate such errors and guarantee correctness for these rules; formal verification plays an important role. Formal verification is a technique used to ensure the correctness of complex software and hardware systems. In this work, an approach to ensure the correctness of the decision rule is introduced. It used CP-Nets formalism to prove decision rules corrects against biasing and mislabelling errors. The decision rules are formally verified by modelling equivalent CP-Net models. The equivalent CP-Net model correctness is ensured using simulations and state-space analysis.

A decision tree, a supervised machine learning algorithm, was applied to induce the classification model. The rule derivation was performed to obtain decision rules from the classification models using the root node to leaf node path traversing. The derived rules are transformed into CP-Nets. The CP-Nets were simulated with colour tokens containing the applicant's academic details to prove the decision rules' correctness and comprehensibility. The absence of errors in the decision rules was guaranteed using state-space analysis. The empirical results showed that the proposed approach improved the correctness of decision rules by integrating formal verification. Therefore, from this research work, it is concluded that the extracted decision rule correctness can be guaranteed by CP-Nets formalism. The research results can be used to improve the higher education institute educational decision-making process. These resultant decision rules can provide applicant background insights for higher educational institutes' academic administration for improving the admission process, curriculum, and learning activities.

In this work, only the decision tree algorithm-based decision rules have been exploited with CP-Nets formalism. The proposed approach has limitations and needs further investigation:

1) The data set only contains the applicant's details of six departments from one higher education institute; for generalized results, the proposed methodology needs to be validated with many higher educational institute data sets.
2) Our proposed approach currently uses only the decision tree-based decision rules. We can exploit other machine learning methods for better-improved decision rules.

We plan to investigate more supervised machine learning techniques and algorithms for decision rule correctness guarantees in future work. In this work, only CP-Nets formalism is explored. In the future, other formal verification techniques like theorem proving need to be investigated for performance improvement of the verification process of decision rules.

ACKNOWLEDGMENT

The authors would like to thank the Directorate of Information Technology, The Islamia University Bahawalpur for providing admission data for academic research purposes.

REFERENCES

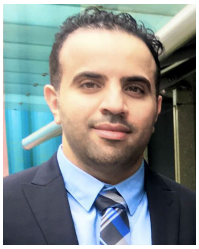
[1] Y. Nieto, V. Gacia-Diaz, C. Montenegro, C. C. Gonzalez, and R. G. Crespo, "Usage of machine learning for strategic decision making at higher educational institutions," *IEEE Access*, vol. 7, pp. 75007–75017, 2019.
[2] S. Maldonado, J. Miranda, D. Olaya, J. Vásquez, and W. Verbeke, "Redefining profit metrics for boosting student retention in higher education," *Decis. Support Syst.*, vol. 143, Apr. 2021, Art. no. 113493.
[3] D. Olaya, J. Vásquez, S. Maldonado, J. Miranda, and W. Verbeke, "Uplift modeling for preventing student dropout in higher education," *Decis. Support Syst.*, vol. 134, Jul. 2020, Art. no. 113320.
[4] S. M. Shuhidan, N. Mastuki, and W. M. N. W. M. Nori, "Accounting information system and decision useful information fit towards cost conscious strategy in malaysian higher education institutions," *Procedia Econ. Finance*, vol. 31, pp. 885–895, Jan. 2015.
[5] A. Y. Noaman and F. F. Ahmed, "ERP systems functionalities in higher education," *Procedia Comput. Sci.*, vol. 65, pp. 385–395, Jan. 2015.
[6] T. Anastasios, S. Cleo, P. Effie, T. Olivier, and M. George, "Institutional research management using an integrated information system," *Procedia Social Behav. Sci.*, vol. 73, pp. 518–525, Feb. 2013.
[7] H. Lounis, T. F. Gayed, and M. Boukadoum, "Using efficient machine-learning models to assess two important quality factors: Maintainability and reusability," in *Proc. Joint Conf. 21st Int. Workshop Softw. Meas. 6th Int. Conf. Softw. Process Product Meas.*, Nov. 2011, pp. 170–177.
[8] J. R. Quinlan *et al.*, "Learning with continuous classes," in *Proc. 5th Austral. Joint Conf. Artif. Intell.*, vol. 92. Singapore: World Scientific, 1992, pp. 343–348.
[9] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.
[10] X. Xu, J. Wang, H. Peng, and R. Wu, "Prediction of academic performance associated with Internet usage behaviors using machine learning algorithms," *Comput. Hum. Behav.*, vol. 98, pp. 166–173, Sep. 2019.
[11] B. Zhang, J. Ren, Y. Cheng, B. Wang, and Z. Wei, "Health data driven on continuous blood pressure prediction based on gradient boosting decision tree algorithm," *IEEE Access*, vol. 7, pp. 32423–32433, 2019.
[12] J. He, H.-J. Hu, R. Harrison, P. C. Tai, and Y. Pan, "Rule generation for protein secondary structure prediction with support vector machines and decision tree," *IEEE Trans. Nanobiosci.*, vol. 5, no. 1, pp. 46–53, Mar. 2006.
[13] N. Alghanmi, R. Alotaibi, and S. M. Buhari, "HLMCC: A hybrid learning anomaly detection model for unlabeled data in Internet of Things," *IEEE Access*, vol. 7, pp. 179492–179504, 2019.
[14] C.-K. Chang, C.-S. Lai, and R.-N. Wu, "Decision tree rules for insulation condition assessment of pre-molded power cable joints with artificial defects," *IEEE Trans. Dielectr. Electr. Insul.*, vol. 26, no. 5, pp. 1636–1644, Oct. 2019.
[15] A. Onan, "Mining opinions from instructor evaluation reviews: A deep learning approach," *Comput. Appl. Eng. Educ.*, vol. 28, no. 1, pp. 117–138, Jan. 2020.
[16] A. Onan, "Sentiment analysis on massive open online course evaluations: A text mining and deep learning approach," *Comput. Appl. Eng. Educ.*, vol. 29, no. 3, pp. 572–589, May 2021.
[17] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, and S. D. Lee, "Decision trees for uncertain data," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 1, pp. 64–78, Jan. 2011.
[18] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," 2019, *arXiv:1901.04966*. [Online]. Available: http://arxiv.org/abs/1901.04966
[19] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3315–3323.
[20] M.-R. Bouguelia, Y. Belaïd, and A. Belaïd, "Identifying and mitigating labelling errors in active learning," in *Pattern Recognition: Applications and Methods*. Cham, Switzerland: Springer, 2015, pp. 35–51.
[21] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 3, pp. 447–461, Mar. 2016.
[22] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2503–2511.
[23] B. J. Taylor and M. A. Darrah, "Rule extraction as a formal method for the verification and validation of neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 5. IEEE, 2005, pp. 2915–2920.

[24] A. Rozinat, R. S. Mans, M. Song, and W. M. P. van der Aalst, "Discovering simulation models," *Inf. Syst.*, vol. 34, no. 3, pp. 305–327, May 2009.
[25] K. Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*, vol. 1. Springer, 2013.
[26] C. A. Petri, "Kommunikation mit automaten," Univ. Bonn, Bonn, Germany, 1962.
[27] R. Milner, M. Tofte, R. Harper, and D. MacQueen, *The Definition Standard ML: Revised*. Cambridge, MA, USA: MIT Press, 1997.
[28] A. J. Clark, "It governance: Determining who decides," *EDUCAUSE Center Appl. Res. Bull.*, vol. 24, pp. 1–13, Nov. 2005.
[29] S.-J. Chan and C.-Y. Yang, "Governance styles in taiwanese universities: Features and effects," *Int. J. Educ. Develop.*, vol. 63, pp. 29–35, Nov. 2018.
[30] M. Guilbault, "Students as customers in higher education: The (controversial) debate needs to end," *J. Retailing Consum. Services*, vol. 40, pp. 295–298, Jan. 2018.
[31] K. Jensen and L. M. Kristensen, *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*, 1st ed. Springer, 2009.
[32] N. Bhargava, G. Sharma, R. Bhargava, and M. Mathuria, "Decision tree analysis on J48 algorithm for data mining," *Proc. Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 6, pp. 1–8, 2013.
[33] S. L. Salzberg, *C4. 5: Programs for Machine Learning*, J. R. Quinlan, Ed. San Mateo, CA, USA: Morgan Kaufmann, 1994.
[34] E. M. Clarke Jr, O. Grumberg, D. Kroening, D. Peled, and H. Veith, *Model Checking*. Cambridge, MA, USA: MIT Press, 2018.
[35] A. H. TCHEMRA, "Tabela de decisão adaptativa na tomada de decisões multicritério. 172 p," Ph.D. dissertation, Tese (Doutorado)-Escola Politécnica da Universidade de São Paulo, São Paulo, Brazil, 2009.
[36] A. Akram, C. Fu, Y. Li, M. Y. Javed, R. Lin, Y. Jiang, and Y. Tang, "Predicting students' Academic procrastination in blended learning course using homework submission data," *IEEE Access*, vol. 7, pp. 102487–102498, 2019.
[37] Y. Yao, Z. Zhang, H. Cui, T. Ren, and J. Xiao, "The influence of student abilities and high school on student growth: A case study of Chinese national college entrance exam," *IEEE Access*, vol. 7, pp. 148254–148264, 2019.
[38] N. Tomasevic, N. Gvozdenovic, and S. Vranes, "An overview and comparison of supervised data mining techniques for student exam performance prediction," *Comput. Educ.*, vol. 143, Jan. 2020, Art. no. 103676.
[39] A. Onan and M. A. Toçoğlu, "Weighted word embeddings and clustering-based identification of question topics in MOOC discussion forum posts," *Comput. Appl. Eng. Educ.*, to be published. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.22252, doi: 10.1002/cae.22252.
[40] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Systematic ensemble model selection approach for educational data mining," *Knowl.-Based Syst.*, vol. 200, Jul. 2020, Art. no. 105992.
[41] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split optimized bagging ensemble model selection for multi-class educational data mining," *Int. J. Speech Technol.*, vol. 50, no. 12, pp. 4506–4528, Dec. 2020.
[42] M. Shahabi, H. Hassanpour, and H. Mashayekhi, "Rule extraction for fatty liver detection using neural networks," *Neural Comput. Appl.*, vol. 31, no. 4, pp. 979–989, Apr. 2019.
[43] J. Sultana, M. U. Rani, and M. A. H. Farquad, "Knowledge discovery from recommender systems using deep learning," in *Proc. Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Nov. 2019, pp. 1074–1078.
[44] J. R. Quinlan, "Generating production rules from decision trees," in *Proc. IJCAI*, vol. 87, 1987, pp. 304–307.
[45] L.-M. Fu, "Recognition of semantically incorrect rules: A neural-network approach," in *Proc. 3rd Int. Conf. Ind. Eng. Appl. Artif. Intell. expert Syst. (IEA/AIE)*, 1990, pp. 1013–1018.
[46] L. Gao, Z. Zhao, L. Qi, Y. Liang, and J. Du, "Modeling the effort and learning ability of students in MOOCs," *IEEE Access*, vol. 7, pp. 128035–128042, 2019.
[47] B. K. Francis and S. S. Babu, "Predicting academic performance of students using a hybrid data mining approach," *J. Med. Syst.*, vol. 43, no. 6, p. 162, Jun. 2019.
[48] V. L. Miguéis, A. Freitas, P. J. V. Garcia, and A. Silva, "Early segmentation of students according to their academic performance: A predictive modelling approach," *Decis. Support Syst.*, vol. 115, pp. 36–51, Nov. 2018.
[49] B. Jia, K. Niu, X. Hou, N. Li, X. Peng, P. Gu, and R. Jia, "Prediction for student academic performance using SMNaive Bayes model," in *Proc. Int. Conf. Adv. Data Mining Appl.* Springer, 2019, pp. 712–725.
[50] K. S. Roy, K. Roopkanth, V. U. Teja, V. Bhavana, and J. Priyanka, "Student career prediction using advanced machine learning techniques," *Int. J. Eng. Technol.*, vol. 7, no. 2, p. 26, 2018.
[51] M.-R. Bouguelia, S. Nowaczyk, K. C. Santosh, and A. Verikas, "Agreeing to disagree: Active learning with noisy labels without crowdsourcing," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 8, pp. 1307–1319, Aug. 2018.
[52] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 1196–1204.
[53] J. Bootkrajang and A. Kabán, "Label-noise robust logistic regression and its applications," in *Machine Learning and Knowledge Discovery in Databases*. Berlin, Germany: Springer, 2012, pp. 143–158.
[54] J. Bootkrajang and A. Kabán, "Learning a label-noise robust logistic regression: Analysis and experiments," in *Intelligent Data Engineering and Automated Learning—IDEAL 2013*. Berlin, Germany: Springer, 2013, pp. 569–576.
[55] C. Scott, G. Blanchard, and G. Handy, "Classification with asymmetric label noise: Consistency and maximal denoising," in *Proc. Conf. Learn. Theory*, 2013, pp. 489–511.
[56] G. Patrini, F. Nielsen, R. Nock, and M. Carioni, "Loss factorization, weakly supervised learning and label noise robustness," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 708–717.
[57] L. Pulina and A. Tacchella, "Challenging SMT solvers to verify neural networks," *AI Commun.*, vol. 25, no. 2, pp. 117–135, 2012.
[58] K. Scheibler, L. Winterer, R. Wimmer, and B. Becker, "Towards verification of artificial neural networks," in *Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen MBMV*. Chemnitz, Germany: Sächsische Landesbibliothek, Mar. 2015, pp. 30–40.
[59] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verisig: Verifying safety properties of hybrid systems with neural network controllers," in *Proc. 22nd ACM Int. Conf. Hybrid Syst., Comput. Control*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 169–178.
[60] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *Proc. Int. Conf. Comput. Aided Verification*. Springer, 2017, pp. 3–29.
[61] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Proc. Int. Conf. Comput. Aided Verification*. Springer, 2017, pp. 97–117.
[62] R. Ehlers, "Formal verification of piece-wise linear feed-forward neural networks," in *Proc. Int. Symp. Automated Technol. Verification Anal.* Springer, 2017, pp. 269–286.
[63] J. Törnblom and S. Nadjm-Tehrani, "Formal verification of input-output mappings of tree ensembles," *Sci. Comput. Program.*, vol. 194, Aug. 2020, Art. no. 102450.
[64] J. Törnblom and S. Nadjm-Tehrani, "Formal verification of random forests in safety-critical applications," in *Proc. Int. Workshop Formal Techn. Saf.-Crit. Syst.* Springer, 2018, pp. 55–71.
[65] O. Bastani, Y. Pu, and A. Solar-Lezama, "Verifiable reinforcement learning via policy extraction," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Montréal, QC, Canada. Red Hook, NY, USA: Curran Associates, 2018, pp. 2499–2509.
[66] I. H. Witten and E. Frank, "Data mining: Practical machine learning tools and techniques with java implementations," *ACM SIGMOD Rec.*, vol. 31, no. 1, pp. 76–77, Mar. 2002.
[67] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.
[68] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *The WEKA Workbench. Online Appendix for 'Data Mining: Practical Machine Learning Tools and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2016.
[69] E. Turban, R. Sharda, and D. Delen, *Decision Support and Business Intelligence Systems*, 9th ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2010.
[70] Y. Nieto, V. García-Díaz, C. Montenegro, and R. G. Crespo, "Supporting academic decision making at higher educational institutions using machine learning-based algorithms," *Soft Comput.*, vol. 23, no. 12, pp. 4145–4153, Jun. 2019.
[71] K. Kennedy, "Formal methods in the verification and validation of simulation models," in *Proc. Spring Simulation Interoperability Workshop, Simulation Interoperability Standards Organization*, Orlando, FL, USA, 2003, pp. 1–5.
[72] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," in *Proc. Int. Conf. Artif. Intell. Statist.* PMLR, 2020, pp. 702–712.

MUHAMMAD NAUMAN is currently pursuing the Ph.D. degree in computer science with The Islamia University of Bahawalpur (IUB), Pakistan. His research interests include formal approaches, machine learning, data mining, artificial neural networks, decision trees, predictive models, and big data analytics.

NADEEM AKHTAR received the master's degree with a specialization in information system architecture from the Institut Universitaire Professionnalisé, University of South Brittany (UBS), Vannes, France, and the Ph.D. degree from the Laboratory VALORIA of Computer Science, UBS, with the highest honour ''Tres Honorable.'' He is currently working as an Assistant Professor with the Faculty of Computing, The Islamia University of Bahawalpur (IUB), Pakistan. He is the recipient of a number of awards, scholarships and research grants, i.e. Study in France 2004 French Embassy scholarship for Master studies, HEC in France, Teaching assistant for ENSIBS=million, research award in year 2014 from The Directorate of Research and Development, The Islamia University of Bahawalpur.

ADI ALHUDHAIF received the bachelor's degree in computer science from King Saud University, the master's and Ph.D. degrees in computer science with a specialization in information security and big data from George Washington University, Washington, DC, USA, and the master's degree in law LLM in Internet law from the University of Strathclyde, Glasgow, U.K., and several IT professional certificates in IT governance, risk management, project management, and more. He is currently an Assistant Professor with Prince Sattam Bin Abdulaziz University.



ABDULRAHMAN ALOTHAIM received the M.Sc. degree in information systems from the University of Maryland, and the Ph.D. degree in information technology from the University of Nebraska. He is currently working as an Assistant Professor of information systems with the College of Computer and Information Sciences, King Saud University. He is also working as a Consultant of Digital Banking with Alinma Bank. His research interests include artificial intelligence, Arabic NLP, machine learning, crowdsourcing, electronic governments, digital transformation, and digital banking.