A Novel Approach to calculate Dominance Based Rough Sets Approximations for dynamic datasets

Author

Uzma Nawaz

00000328425

Supervisor

Dr. Usman Qamar

DEPARTMENT OF COMPUTER AND SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

February, 2024

Annex A

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by NS **Uzma Nawaz** Registration No. **00000328425**, of College of E&ME has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the thesis.

Signature : [illegible]

Name of Supervisor: **Dr Usman Qamar**

Date: 27-02-2024

Signature of HOD: [illegible]
(Dr Usman Qamar)
Date: 27-02-2024

Signature of Dean: [illegible]
(Brig Dr Nasir Rashid)
Date: 27 FEB 2024

A Novel Approach to calculate Dominance Based Rough Sets Approximations for dynamic datasets

Author

Uzma Nawaz

00000328425

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Computer Engineering

Thesis Supervisor

Dr. Usman Qamar

[illegible]

Thesis Supervisor Signature: ____________________________________

DEPARTMENT OF COMPUTER AND SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

February, 2024

Dedicated to my exceptional parents, sister, brother and friends whose tremendous support and cooperation led me to this wonderful accomplishment

Acknowledgement

First of all, we are grateful to **ALLAH** Almighty who is beneficent and most merciful. Who made the universe with the possibility of glory, balance and amicability, with normality and gave the capacities to human to develop. **ALLAH** Almighty has blessed us with beloved parents and helpful friends who consistently contributed towards our studies.

I am incredibly grateful to my wonderful parents who reared me when I was unable to walk and who have supported me throughout my life in every aspect.

I'm really thankful to my honorable Supervisor Dr. Usman Qamar for his help and assistance. It is a matter of great pleasure to express our regards to worthy Supervisor, for his masterly advice, encouragement and guidance. With his help we are able to complete our monotonous work.

Additionally, I would like to express my gratitude to Dr. Summair Raza for his outstanding assistance and cooperation. He always had an answer when I was stuck on something. I wouldn't have been able to finish my thesis without his assistance. I am grateful for his assistance and patience throughout the entire thesis.

For their assistance and cooperation, I also want to thank Drs. Wasi Haider and AP Jahan Zeb from my GEC committee.

Finally, I would want to offer my gratitude to everyone who has helped my research in any way.

Abstract

Effective data analysis of big datasets is a challenge and when data changes over time, new feature values are added, as a result, its complexity increases. An advanced approach of rough sets theory known as the Dominance-based Rough Set Approach (DRSA) is a powerful mathematical tool for identifying meaningful information in preference-ordered datasets. Data analysis using DRSA is primarily based on the lower and upper approximations calculation, which are very expensive to compute. It mainly used resources i.e., time execution and memory consumption. When the data changes over time, approximation sets must be recalculated. As a result, calculations that are repeating, increase the cost of approximation's computation in the real-time domain. The proposed approach computes approximations for increasing object values with time. Results were compared with the conventional approach using UCI publicly available datasets. When compared to the usual method, our suggested approach updated approximations in less time, with an average reduction of 99.2%.

Index Terms- Rough Set Theory, Dominance Based Rough Sets Approximation, Dynamic Dataset, KD Tree, Dynamic Update, Dynamic System

Table of Contents

List of Figures

List of Tables

CHAPTER 1 : INTRODUCTION

The concept of data is constantly changing and evolving in our daily lives. Data complexity refers to the level of difficulty in examining and handling data [1]. Due to the rapid expansion of digital information, data complexity become a serious challenge. Factors that affect complexity of data are its expansion, growth, velocity and variety. When large amount of data generated from its source, it's way more difficult to handle, analyze, preprocess and computation analysis [2]. An algorithm can become affected by irrelevant, redundant, and noisy features, which will have a negative impact on efficiency, accuracy and computational cost.

Dealing with large data requires advance techniques, tools and expertise to find meaningful information for decision making process [3]. So, there is a need to select relevant features that can help algorithm to work more effectively and efficiently.

Relevant features lead to data processing more accurately. But as the quantity and variety of datasets increase with time, it contains irrelevant and redundant features most likely noisy data. As the quantity of data increases, the size also increases. It will affect the model's accuracy and efficiency very badly. With the help of the feature selection techniques, we can eliminate irrelevant and redundant features.

It also helps in improving the model's efficiency, and performance and lowers its computational expense. It reduces overfitting and improves accessibility. To assess the relevance and value of features, various techniques such as statistical testing, correlation analysis, and regularization procedures are used.

Effective feature selection can result in more efficient and accurate models which allow for better decision-making and data insights [4]. With the feature selection method, we can remove irrelevant features and make the algorithm easier to understand and debug. Also, we can increase the model's performance and enable the model to train fast.

1.1 Feature selection

The process of selecting the most reliable, non-redundant, and relevant characteristics to add in a model is known as feature selection [5]. As the number and diversity of datasets expand over time, it is critical to gradually minimize their size. The basic goals of feature selection are to improve the performance of a predictive model while also lowering its computing cost.

The act of selecting the most important features to input into algorithmic methods for machine learning is known as feature selection, and it is one of the primary elements of feature design [6]. By deleting redundant or unneeded features and reducing the number of features that are most critical to the predictive model.

1.1.1 Advantages of Feature Selection

There are many advantages of feature selection. Some of them are listed below:

1. It removes irrelevant features
2. It makes algorithm easier to understand and debug
3. It increases the performance of model
4. It makes training fast

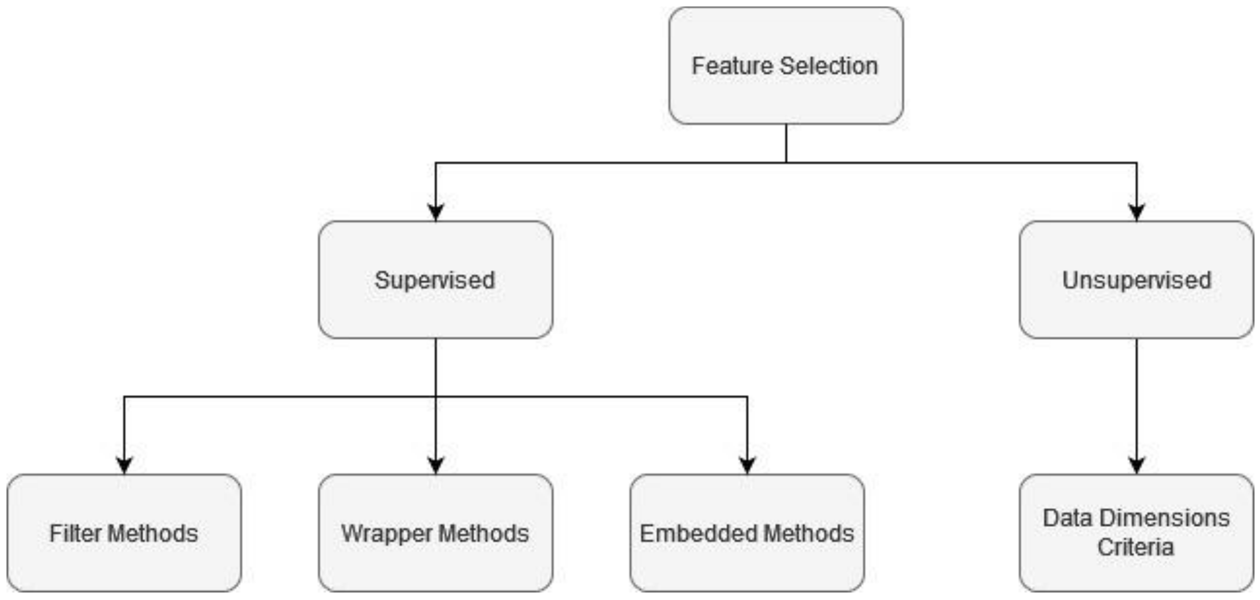


Figure 1 : Generic Process of feature selection

1.2 Methods of Feature selection

There are two types of feature selection algorithms:

1.2.1 Supervised

This strategy may be used on labeled data to identify relevant features [7] and improve the efficacy of supervised models [6] such as regression and classification. This includes SVM, decision trees, and linear regression. Supervised techniques also include filter methods, wrapper methods, and embedded methods.

1.2.1.1 Filter Methods

In contrast to cross-validation performance, filter algorithms pick features based on statistics [9], [10]. A predetermined measure is used to identify superfluous properties and perform recursive feature selection. Multivariate filter approaches assess the overall relevance of the features while identifying redundant and unnecessary features [11], in contrast to univariate filter methods, which create a specific ordering list of features to help with the final pick of feature subsets.

1.2.1.2 Wrapper Methods

Wrapper feature selection techniques treat feature selection as a search issue by preparing, evaluating, and comparing a particular feature combination to another [12], [13]. This method makes it easier to identify possible interactions between variables. Wrapper techniques focus on feature subsets to enable the selection algorithm provide higher-quality outcomes [14]. Boruta and Forward feature selection are two popular examples.

1.2.1.3 Embedded Methods

In this methods, classification and selection of features are carried out concurrently as part of the learning algorithm by integrating the feature selection machine learning algorithm [15]. During every iteration of the model training process, the most important characteristics are carefully extracted [16]. Embedded techniques include random forest feature selection, decision tree feature selection, and LASSO feature selection.

1.2.2 Unsupervised

This method can be applied to unlabeled data that can be used [17] with these methods. Because there are no clearly defined data labels in unsupervised, it is difficult to evaluate the effectiveness of an algorithm [18] The algorithm divides a dataset into various categories after discovering a deep structure in the data on its own. The algorithm is frequently selected in accordance with business objectives. Examples include Principal Component Analysis, Hierarchical Clustering, and K-Means Clustering.

1.2.2.1 Data Dimensions Criteria

The machine learning (ML) technique known as dimensionality reduction is used to identify patterns in data and solve complex computational problems [19]. This approach uses a collection of algorithms transform complex input data to a space of low dimensions., thereby reducing the number of input variables in a dataset [20]. When working with visual and audio data involving speech, video, images, or text, as well as when condensing datasets to better fit a predictive model, dimension reduction is useful.

1.3 Rough Set Theory

Rough set theory [21] is a mathematical approach developed by Zdzislaw Pawlak in 1982 that is used to manage [22] unclear, irrelevant, incomplete, imprecise, and noisy data for decision making and data analysis. RST provides a number of data structures to represent real-world data, including Information Systems, Decision Systems, and Approximations [23].

Based on the indiscernibility connection, rough set theory splits data into two different and equivalent types. By indiscernibility, we imply that a data collection cannot be recognized by a certain set of characteristics. Rough set theory allows for the identification of relevant traits while deleting those that are redundant or unnecessary [26]. RST offers a variety of data types to represent data from real-world situations such as Information Systems, Reasoning Systems, and Approximate values. It offers a structured framework for knowledge discovery, feature selection, and rule induction, making it particularly useful in data mining, machine learning, and expert systems. Rough set theory has applications in a wide range of domains, including finance, medicine, and pattern recognition. Because of its ability to deal with ambiguity and missing or inaccurate data, it is beneficial in circumstances where traditional approaches may fail.

The theory provides a solid foundation for analyzing and interpreting big data, enabling the extraction of important data and relevant qualities. Its adaptability and agility make it a vital tool in artificial intelligence, aiding with the creation of statistical analysis and support systems for decision-making.

Rough set theory, like probability, statistics, entropy, and Dempster-Shafer theory, is one of several methodologies that may be used to analyze uncertain systems. However, this method does not work for preferred ordering.

1.4 Dominance Based Rough Set Approach (DRSA)

To address this issue, Greco, Matarazzo, and Sowiski [24] suggested dominance-based rough set theory, a modified version of rough set theory. DRSA [25] uses the dominance connection to replace the indiscernibility connection. Theory of rough set is a strong paradigm for coping with unclear data and inadequate expertise in data analysis and decision-making strategies based on dominance. It incorporates object dominance relations into the usual rough set theory.

Objects are compared based on their preference order in this theory, and dominance connections are built to assess the relative value of items. Rough set theory based on dominance provides for a deep understanding of the interactions between items and features based on preference order. It provides a versatile and effective method for feature selection, classification, and decision-making tasks, particularly when data is few or imprecise.

The fundamental benefit of utilizing rough set theory based on dominance is that it can handle complicated and unpredictable material without requiring precise numerical values. The theory can represent the inherent complexity and incomplete data in real-world by studying preference order. This makes it especially effective in sectors where data is likely to be inadequate, noisy, or open to human interpretation. DRSA has found applications in a variety of disciplines, that includes finance, healthcare, and engineering, where decision-making under uncertainty is important.

DRSA calculates lower and upper approximations based on preference order between objects. To achieve this, objects are compared according to their order of preference. The computational complexity develops because dominance-based rough set theory considers various qualities and their combinations. As dataset increases its attribute set also increases, as a result the number of dominance comparison also grows. This cause consumption of resources such as time and memory.

Approximation sets must be recalculated when the data evolve over time. Therefore, Repeated computations raise the computational price of approaches in the current time realm. To avoid this problem, we devised a solution in which the method computes approximations for object values that increase over time.

The conventional definition of DRSA approximations is finding objects that fit into the greatest and least value ordered choice classes are required. The stage is computationally costly for bigger datasets. Rough approximation computations are complex and time-consuming, which contributes to DRSA's slow performance. Our research methodology lowers the quantity of RAM required and the processing time.

For a single approximation, the traditional DRSA model repeatedly compares each instance and its attributes with the complete dataset. In this research paper, we provide an optimized method to improve efficiency in updating approximation sets and to decrease time complexity. All estimates were correctly updated using the proposed approach, which also avoided unnecessary iterations across the whole dataset.

1.5 Motivation

DRSA is a modified version of rough set theory. It easily handles difficult data. It is used by many applications for a variety of data mining tasks. It is advance concept but reduction in computational difficulty is still required. A lot of researchers work to minimize the computational complexity and cost.

1.6 Problem description

DRSA calculates greatest and least value ordered choice classes based on ranking of preferences between objects. This is done by comparing objects based on their ranking of preferences. The computational difficulty develops because dominance-based rough set theory considers various qualities and their combinations. As dataset increases, its attribute set also increases, as a result the number of dominance comparison also grows. This cause consumption of resources such as time and memory.

Approximation sets must be recalculated when the data evolve over time. Therefore, the computational cost of approximations in the real-time domain is increased by repetitive calculations. To avoid this problem, we devised a solution in which the method computes approximations for object value that increase over time.

1.7 Aims and Objectives

The primary goals are:

- To reduce the computational burden of modelling and improve the performance of a predictive model.
- To determine the upper and lower bounds of approximations for dynamic datasets in which object values vary over time.
- To reduce execution time and memory consumption.

1.8 Scope

The standard description of DRSA approximations is that finding items that fit into both the greatest and least prefer ordered decision classes is required. This stage is computationally intensive for bigger datasets. Rough approximation computations are laborious and time-consuming, which contributes to DRSA's sluggish performance. Our methodology lowers the amount of RAM used and the observed execution time.

The old DRSA model [27] compares every instance and its attributes with the entire dataset numerous times for a single approximation. In this paper we come with optimized approach to reduce time complexity and to update approximation sets more efficiently. The proposed approach effectively updated every approximation without going through the whole dataset in needless rounds.

Our proposed model reduces cost of DSRA algorithm, it focuses on improving less memory consumption, improves time complexity and sticks to the absolute accuracy. We have tested our proposed method's effectiveness using ten UCI datasets and compared its performance to that of traditional algorithm as well as parallel and incremented approaches.

The results demonstrated that our suggested strategy decreased execution time by almost 99.2%. The main objective of our study is to show how the proposed method can be used in real-world problems such as medical diagnostics, using less processing and storage space to perform behavior analysis, prediction, categorization, and making choices on huge datasets.

1.9 Research Contributions

We contributed following things in our research:

- Reduce computational cost of DRSA algorithm
- Focus on improving less memory consumption
- Improve time complexity
- Stick to absolute set accuracy

1.10 National Needs

The primary goal of research is to gain knowledge in order to better build our nation. We must link research with national development in order to build a society, polity, and economy that eliminates poverty, unemployment, and inequality, along with other things. The main objective of our study is to show how the proposed method can be used in real-world problems such as medical diagnostics, using less processing and storage space to perform behavior analysis, prediction, categorization, and making choices on huge datasets with prefer choice attributes.

1.11 Applications

Pawlak proposed the revolutionary RST in 1982, which was a milestone in the history of non-statistical data mining approaches [21]. Building on this basis, the DRSA evolved as a modified version which demonstrate amazing ability to handle complex data structures [24]. Its adaptability has resulted in extensive usage in a variety of disciplines, including financial forecasting, image segmentation, classification tasks, and fault identification. Furthermore, DRSA is used in data exploration to facilitate the extraction of useful insights from difficult datasets. Furthermore, it is an effective tool for developing exact decision rules, which improves decision-making processes in a variety of scenarios. Notably, DRSA plays an important part in multi-criteria decision aid systems which allow it for the selection of optimal investment projects among competing options. DRSA remains the foundation of modern data mining procedures, because to its broad applicability and excellent performance.

1.12 Thesis Structure

Following is the overall thesis structure:

- **Chapter 2: Dominance Based Rough Set Approach (DRSA):**
  Rough Set Theory's foundational concepts are covered in Chapter 2. Additionally, it analyses rough-set theory and gives examples of both its strengths and weaknesses.
- **Chapter 3: Literature Review**
  In this chapter we have discussed some preliminary concepts regarding DRSA
- **Chapter 4: Challenges in computation:**
  In this chapter we discussed the challenges in computation of DRSA and its algorithm.
- **Chapter 5: Proposed methodology:**
  In this chapter, we have put forth a computationally efficient algorithm to replace the traditional upper and lower approximation algorithms used in DRSA.
- **Chapter 6: Results and analysis:**
  Results are produced in this chapter after our suggested solution has been tested on various datasets. Following that, an analysis and conclusion are drawn based on the findings of the experiment.
- **Chapter 7: Conclusion and future work:**
  In this chapter we concluded our work and additional future work is provided.

CHAPTER 2 : DOMINANCE BASED ROUGH SET APPROACH

In this chapter, we will discuss major concepts regarding DRSA. In 2.1 main concept is elaborated. After that in 2.2, related terms are explained. Further in 2.3, dominance relation is discussed. In 2.4, class concept is described. In 2.5, approximation calculation steps are shown and in last section 2.6, how to find reducts and core using upper and lower approximation is elaborated.

2.1 Concepts

Data mining is done by classification, which is a process of extracting and discover the vital knowledge from large datasets. With the help of classification method [5],[6], data points are separated into different classes.

It is used to organize data and improve quality of datasets. But there is a need to improve classification process. In machine learning, feature selection algorithms are applied on them to select relevant features which then further use in classification. There are many algorithms used for feature selection. RST is one of them.

Dataset attributes are of many types. Rough sets work on discrete-valued attributes having independent values and produces good results [21]. But some attributes have ordered values called preference value. These attributes are known as criteria and usually, they have to do with economic matters, like financial or marketing data. [24]. These attributes are different from independent attributes, where rough set couldn't apply on it.

There is a need to make a decision or to do classification with certain criteria. There are objects set that are examined by a certain criterion. When multiple attributes are used to describe an object, multi-criteria decision analysis is used [27].

These attributes have assigned criterion values that belongs to specific class. For example, home appliance has its specific model number and company name. It's country and color have no use. Rough sets can't apply on such attributes. The relationship between attributes known as a monotonic relation occurs when one attribute depends on another. In monotonic relationships, the type of dependence can either be directly proportional or inversely proportional. In a monotonic relation, four types of relationships are taken into account.

- Increase in one attribute increases the other attribute.
- Increase in one attribute decreases the other attribute.
- Decrease in one attribute increases the other attribute.
- Decrease in one attribute decreases the other attribute.

Some examples of monotonic relation are as follows:

- If a person is paid for working at a regular rate, the money that he earned is directly proportional to the time.
- The number of construction workers increases, less time will require to finish the project.
- As the diameter of the circle increases its circumference increases.

Greco, Sowiski, and Matarazzo introduce DRSA to handle properties that are ordered by preference, multi criteria decision analysis, and monotonic relationships [24]. In DRSA, the dominance relation takes place of the indiscernibility relation. Important features are chosen and classification rules are generated with the aid of DRSA.

2.2 Related terms

The dataset is regarded as a decision table in traditional RST. Decision table consist of finite number of elements called universe denoted by U. Having two kinds of attributes i.e., conditional attribute (C) and decision attribute (D). A decision system is described as:

Table 1: Decision Table

Universe	**English**	**Mathematics**	**Grades**
X1	B	B	Good
X2	C	B	Good
X3	B	C	Good
X4	A	B	Very Good
X5	A	A	Excellent
X6	A	B	Very Good

For simplicity, we considered numbers as Excellent = 1, Very Good =2, Good =3 for decision attribute. And for conditional attribute we considered A=1, B=2 and C=3.

$$IS = (U, Q, V, f) \quad (1)$$

U in this case denotes a limited collection of objects (universe), Q represents limited set of criteria $Q = (C \cup D)$, that contains conditional and decision attributes, V = $U_{q \in Q} V_q$ where V_q is the collection of criterion q values. f represents function of $f(x, q)$ which designates a specific value V_q to an object x for attribute q.

DRSA analyses the dominance relation to take into account the objects' preference order. The following is a definition of an object's dominance relationship:

DRSA makes advantage of the prioritized attributes dominance connection to locate reductions and the core. Because P is a valid subset of C, it is employed in this procedure and is subject to a dominance relation. two different kinds of dominance relationships are.

- Dominance positive relation:

$$D_{p^-}(x) = \{y \in \cup : xD_p y\} \quad (2)$$

- Dominance negative relation:

$$D_{p^+}(x) = \{x \in \cup : yD_p x\} \quad (3)$$

DRSA decision union classes further classified into two types.

- Upward Union of classes $Cl_t^{\geq}$: it includes objects greater than preference order t.

$$Cl_t^{\geq} = U_{s \geq t} Cl_s \quad (4)$$

- Downward Union of classes s $Cl_t^{\leq}$:
- it includes objects less than preference order t.

$$Cl_t^{\leq} = U_{s \leq t} Cl_s \quad (5)$$

Whenever a new object is inserted, the approximations are updated.

We assumed that a new set of objects would be used to further clarify the implementation of our suggested strategy $U^{+} = \{\boldsymbol{X7}, \boldsymbol{X8}, \boldsymbol{X9}\}$ details of the conditional and decision attribute are provided in the decision system as **{Universe, English, Mathematics, Grade}:**

- **(X7, C, B, Good)**
- **(X8, B, A, Very Good)**
- **(X9, A, A, Excellent)**

Take new objects X7, X8, X9 into consideration. In order to update the approximation sets, we first update the decision classes and the corresponding groups arranged by greater or lower preference. Consider Table 1, we set object X3 as origin and P = {English}. Then, using equation 2 and 3 we have,

$$D_{p^-}(x) = \{X1, X2, X3, X7\}$$

And

$$D_{p^+}(x) = \{X1, X3, X4, X5, X6, X8, X9\}$$

We have taken preference order t = 2. The set of class union greater and equal to $Cl_t^{\geq}(x)$ have a value set:

$$Cl_t^{\geq}(x) = \{X4, X5, X6, X8, X9\}$$

That is these objects either belonging to class ‘Very Good’ or class ‘Excellent’.

Similarly, $Cl_t^{\leq}(x) = \{X1, X2, X3, X4, X6. X7, X8\}$

This means these objects either belonging to class ‘Very Good’ or less preferred class ‘Good’.

Now, if we change preference order t=1:

$Cl_t^{\geqq}(x) = \{X5, X9\}$

$Cl_t^{\leqq}(x) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

2.3 Approximations

The essential idea in finding reducts and core is approximation. Approximations are computed to determine whether or not an item belongs to a class of decision attribute. Two different categories of approximations exist. In dominance based rough set theory we have two kinds of approximations; one is Lower approximation and other is Upper approximation.

2.3.1 Lower Approximations

Lower approximation in RST defines the set of objects that must belong to a decision class based on the given attributes. Given in DSRA, all the objects that will unquestionably belong to $Cl_t^{\geqq}(x)$ are specified by the P⊆C P-lower approximation. Similarly, all the objects that unquestionably belong to $Cl_t^{\leqq}(x)$ will be included in the P-lower approximation of $Cl_t^{\leqq}(x)$. A subset is required to compute lower approximation. If every element of one set (A) is present in the other set (B), then set A is a proper subset of set B. For example, consider two sets

A = {9, 10, 20, 30}

B = {2, 3, 9, 10, 20, 30}

A is a proper subset of set B because it has all the elements present in set B. Lower Approximation is calculated by these expressions. Mathematically it can be written as:

$For\ Cl_t^{\geqq}(x)$:

$$\underline{P}(Cl_t^{\geqq}) = \{x \in U: Dp^{+}(x) \subseteq Cl_t^{\geqq}\} \qquad (6)$$

For $Cl_t^{\leqq}(x)$:

$$\underline{P}(Cl_t^{\leqq}) = \{x \in U: Dp^{-}(x) \subseteq Cl_t^{\leqq}\} \tag{7}$$

Calculation of Lower approximation is determined by three steps, taking example of above table 1 for $\underline{P}(Cl_t^{\geqq})$ i.e.,

Step 1:

In this step, we calculated the objects belongs to $(Cl_t^{\geqq})$ union class. In our example we have calculated $\underline{P}(Cl_t^{\geqq})\ for\ t = 2$ which is class 'Very Good'.

$Cl_t^{\geqq}(x) = \{X4, X5, X6, X8, X9\}$

Step 2:

In this step, we have found Dp^{+} for each object identified by $Cl_t^{\geqq}(x)$.

$For\ X4: Dp^{+}(X4) = \{X4, X5, X6, X9\}$

$For\ X5: Dp^{+}(X5) = \{X5, X9\}$

$For\ X6: Dp^{+}(X6) = \{X4, X5, X6, X9\}$

$For\ X8: Dp^{+}(X8) = \{X5, X8, X9\}$

$For\ X9: Dp^{+}(X9) = \{X5, X9\}$

The above-mentioned example shows that we must calculate Dp^{+} for each object, which is time consuming and computationally very expensive. If we talk about large dataset or the dataset that changes with time, calculation Dp^{+} of each object by iterating for the whole dataset consumes a lot of time and memory consumption. That will increase computational cost significantly.

Step 3:

In this step which is actually last step of lower approximation calculation, we need to find subset. Lower approximation will include step 2's recognized sets, which are subsets of step 1's identified sets.

$\underline{P}(Cl_t^{\geq}) = \{X5, X9\}$

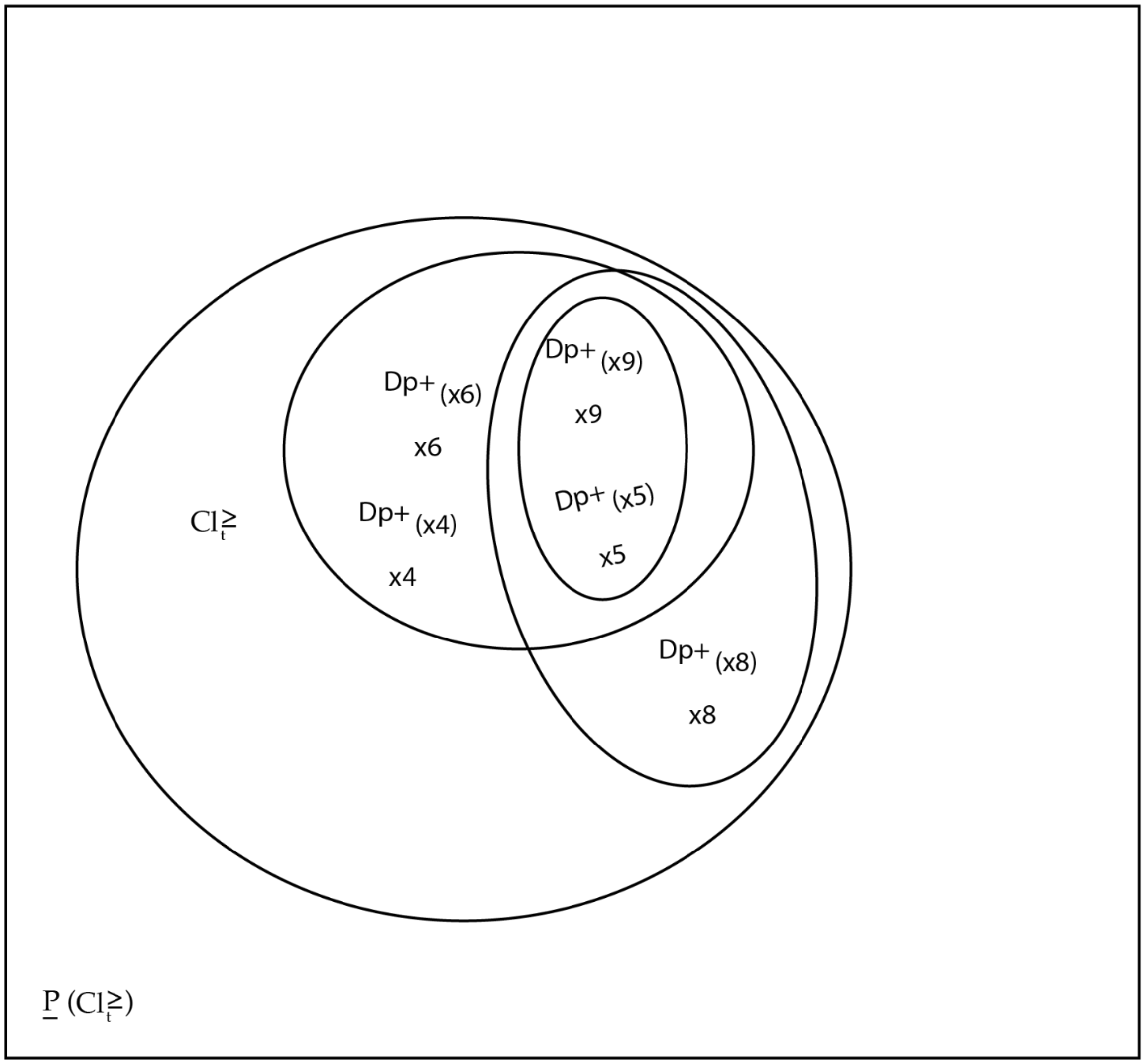


Figure 2: Subset property for lower approximation $\underline{P}(Cl_t^{\leq})$

Repeat above mentioned three steps to get $\underline{\boldsymbol{P}}(\boldsymbol{Cl_t^{\leq}})$

Step 1:

In this step, we calculated the objects belongs to $(Cl_t^{\leq})$ union class. In our example we have calculated $\underline{P}(Cl_t^{\leq})\ \ for\ t = 2$ which is class 'Very Good'. $Cl_t^{\leq}(x) = \{X1, X2, X3, X4, X6, X7, X8\}$

Step 2:

Calculate Dp^{-} for each individual object found in step 1

$For\ X1:\ Dp^{-}(X1) = \{X1, X4, X5, X8, X9\}$

$For\ X2:\ Dp^{-}(X2) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$

$For\ X3:\ Dp^{-}(X3) = \{X3, X4, X5, X6, X8, X9\}$

$For\ X4:\ Dp^{-}(X4) = \{X4, X5, X6, X9\}$

$For\ X6:\ Dp^{-}(X6) = \{X4, X5, X6, X9\}$

$For\ X7:\ Dp^{-}(X7) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$

$For\ X8:\ Dp^{-}(X8) = \{X5, X8, X9\}$

Step 3:

This step gave us lower approximation subset as shown in figure below.

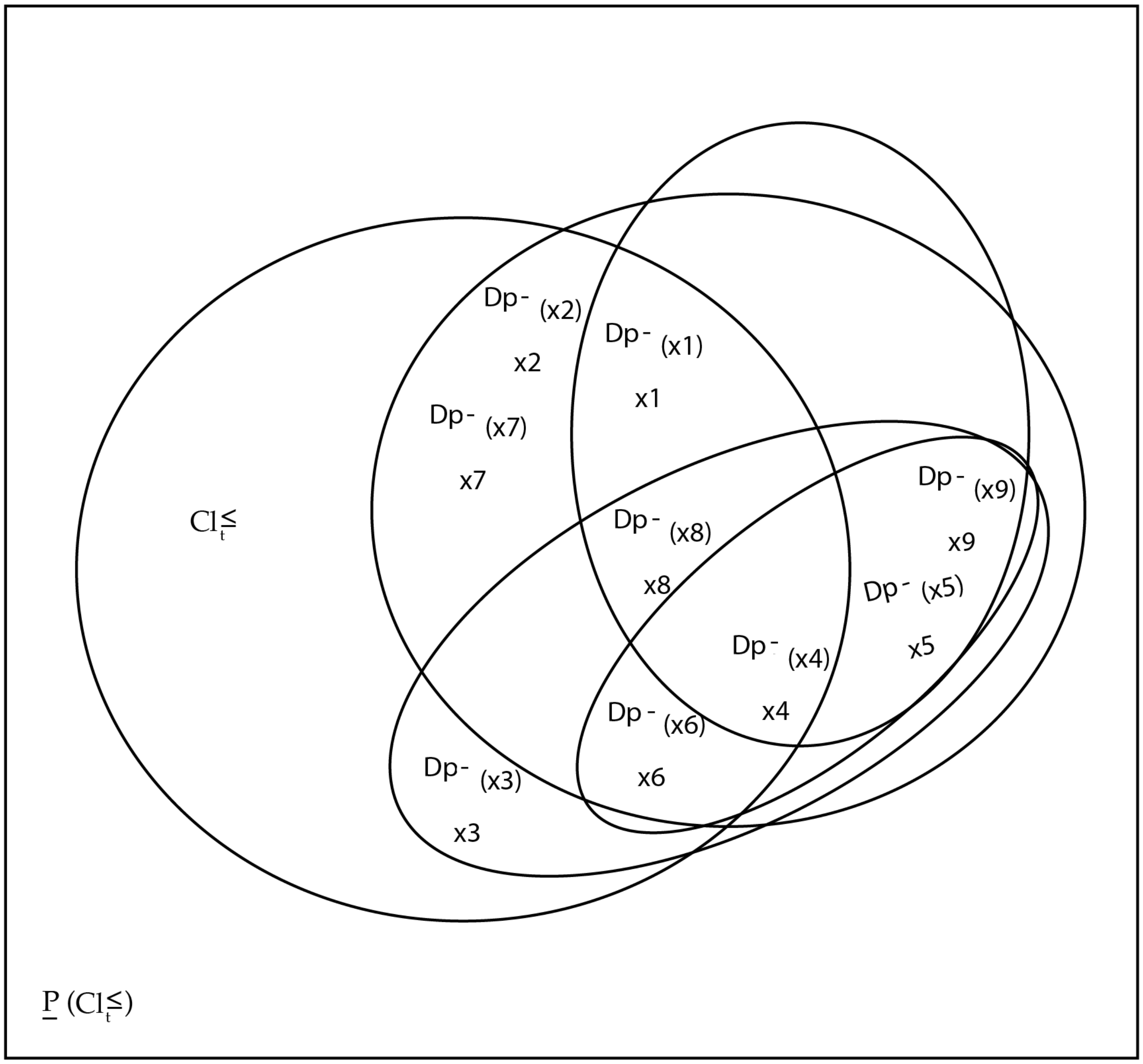


Figure 3: Subset property for lower approximation $\underline{P}(Cl_t^{\leq})$

2.3.2 Upper Approximations

In the conventional RST-based method, the upper approximation determines the set of items that could possibly correspond to the notion X. The set of objects in DSRA that might may be a part of the union of classes $Cl_t^{\geq}(x)$ is defined by the P-upper approximation of $Cl_t^{\geq}(x)$ for P⊆C. Similar to this, the set of objects that relate to the union of classes $Cl_t^{\leq}(x)$ is defined by the P-upper approximation of $Cl_t^{\leq}(x)$. An intersection is required to compute upper approximation. The group of elements that are part of both sets A and B can be found at their intersection. It is denoted as A∩B. For example, consider two sets

A = {9, 10, 20, 30}

B = {2, 3, 9, 10, 20, 30}

A∩B = {9, 10, 20, 30}

Upper Approximation is calculated by these expressions. Mathematically it can be written as:

$For\ Cl_t^{\geq}(x)$:

$$\bar{P}(Cl_t^{\geq}) = \{x\ \in U: Dp^{-}(x) \cap\ Cl_t \geq \neq \emptyset\} \tag{8}$$

For $Cl_t \leq (x)$:

$$\bar{P}(Cl_t^{\leq}) = \{x\ \in U: Dp^{+}(x) \cap\ Cl_t \neq \emptyset\} \tag{9}$$

Calculation of Upper approximation is also determined by three steps, taking above table 1 as an example for $\bar{P}(Cl_t^{\geq})$ i.e.,

We calculated upper approximation with preference order t=2.

Step 1:

In this step, just like step 1 of lower approximation, we again identify the class union. The column of decision attribute is traversed. Items are stored in an array for further processing if their decision label matches the class. We again calculated the objects belongs to $(Cl_t^{\geq})$ union class. In our example we have calculated $Cl_t^{\geq}(x) = \{X4, X5, X6, X8, X9\}$

Step 2:

We calculated $D_{p^{+}}$ of each individual object of $Cl_t^{\geq}(x)$ found in step 1.

$For\ X4:\ Dp^{-}(X4) = \{X1, X2, X3, X4, X6, X7\}$

$For\ X5: \ Dp^{-}(X5) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

$For\ X6: \ Dp^{-}(X6) = \{X1, X2, X3, X4, X6, X7\}$

$For\ X8: \ Dp^{-}(X8) = \{X1, X2, X3, X7, X8\}$

$For\ X9: \ Dp^{-}(X9) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

Step 2 has significantly worsen performance because we must locate objects that dominate each object. To accomplish this, the dataset must be traversed entirely for each object.

Step 3:

In this step, we need to find intersection set. Upper approximation will include the sets identified in step 2 that are intersection of the sets identified in step 1.

$\bar{P}(Cl_t^{\geq}) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

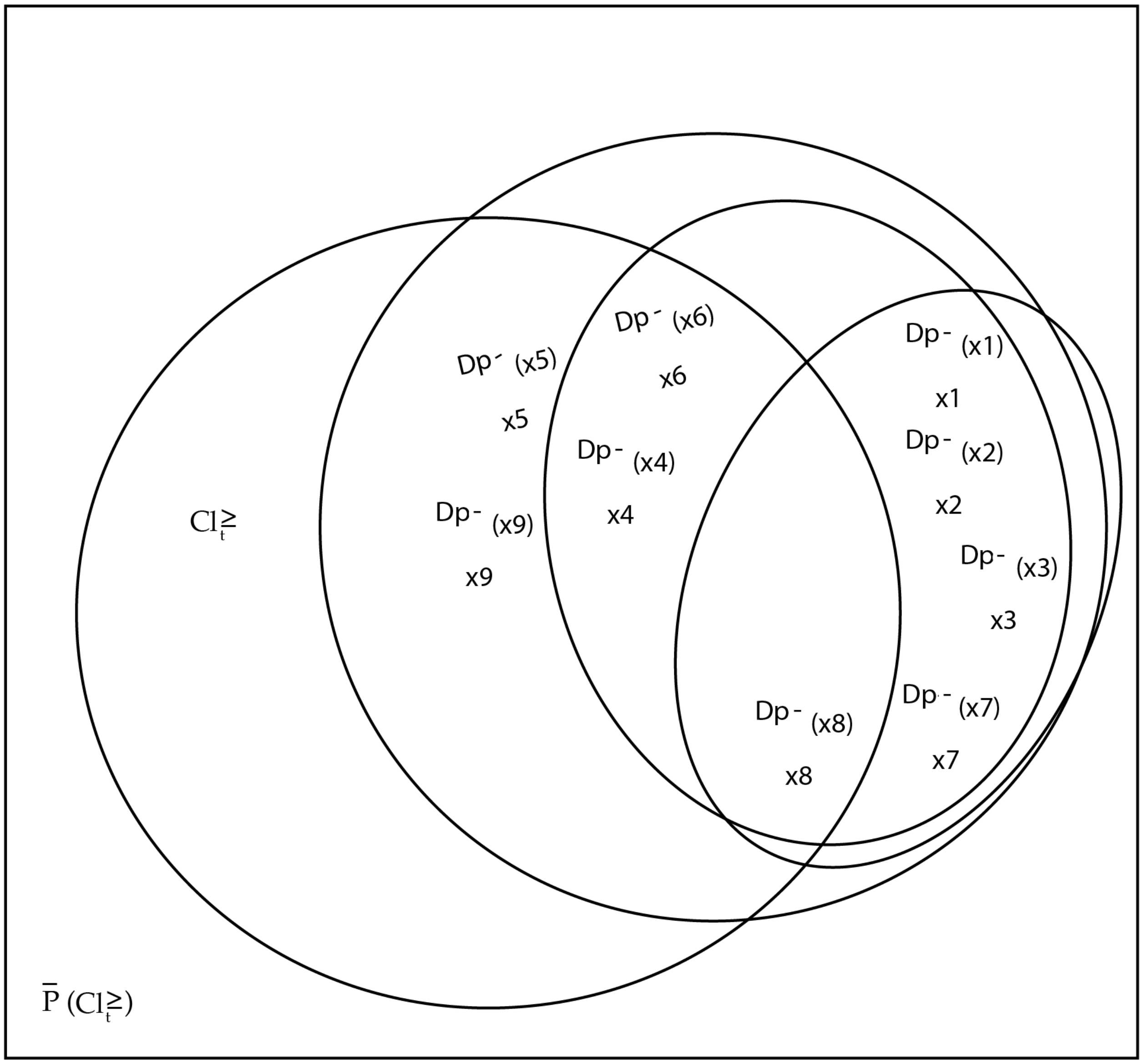


Figure 4: Intersection property for upper approximation $\overline{P}(Cl_t^{\leq})$

Repeat above mentioned three steps for $\overline{\boldsymbol{P}}(\boldsymbol{Cl_t^{\leq}})$

Step 1:

In this step, we've calculated the objects belongs to $(Cl_t^{\leq})$ union class. $Cl_t^{\leq}\,(x) = \{X1, X2, X3, X4, X6, X7, X8\}$

Step 2:

Calculate Dp^{-} for each individual object found in step 1

$For\ X1:\ Dp^{+}(X1) = \{X1, X4, X5, X6, X8, X9\}$

$For\ X2:\ Dp^{+}(X2) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$

$For\ X3:\ Dp^{+}(X3) = \{X1, X4, X5, X6, X8, X9\}$

$For\ X4:\ Dp^{+}(X4) = \{X4, X5, X6, X9\}$

$For\ X6:\ Dp^{+}(X6) = \{X4, X5, X6, X9\}$

$For\ X7:\ Dp^{-}(X7) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$

$For\ X8:\ Dp^{+}(X8) = \{X5, X8, X9\}$

Step 3:

With this step we have got our upper approximation set

$\bar{P}(Cl_t^{\leq}) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

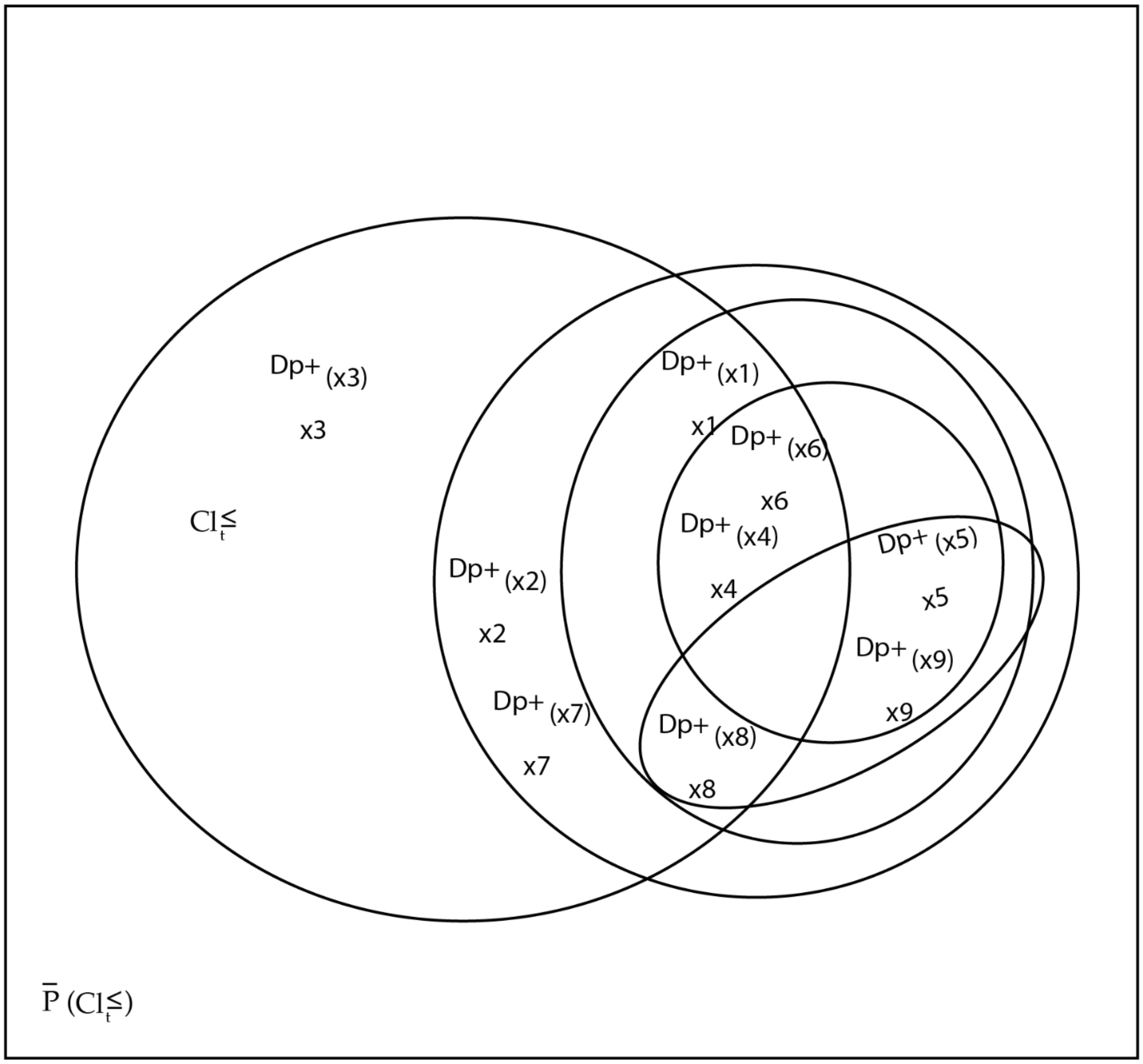


Figure 5: Intersection property of upper approximation $\overline{P}(Cl_t^{\leq})$

Conventional DRSA approach's steps to calculate lower and upper approximations. Conventional method is expensive and time consuming. These steps can apply on a smaller dataset but for large dataset or the dataset that changes with time this traditional approach is not suitable.

2.4 Quality of Approximation and reducts

The following criteria should be met following the discovery of lower and upper approximations.

$$\underline{P}(Cl_t^{\leq}) \subseteq Cl_t^{\leq} \leq \subseteq \bar{P}(Cl_t^{\leq}) \tag{10}$$

$$\underline{P}(Cl_t^{\geq}) \subseteq Cl_t^{\leq} \geq \subseteq \bar{P}(Cl_t^{\geq}) \tag{11}$$

Doubtful regions, also known as P-boundaries, are established using upper and lower approximation. Boundaries are useful for locating reductions.

$$Bn_p(Cl_t^{\leq}) = \overline{P}(Cl_t^{\leq}) - \underline{P}(Cl_t^{\leq}) \tag{12}$$

$$Bn_p(Cl_t^{\geq}) = \overline{P}(Cl_t^{\geq}) - \underline{P}(Cl_t^{\geq}) \tag{13}$$

The degree of approximation quality is expressed as a ratio $r_p(Cl)$. It is denoted mathematically as:

$$r_p(Cl) = |U - (((U_{t\in T}\, Bn_p(Cl_t^{\geq})) \cup ((U_{t\in T}\, Bn_p(Cl_t^{\geq})))|/|U| \tag{14}$$

The reduct of C is known as $P \subseteq C$ for each subset where $r_c(Cl) = r_p(Cl)$ and is denoted as $RED_{Cl}(P)$. A single decision table may have many reducts that exit. Core is the intersection point of all reductions.

CHAPTER 3 : LITERATURE REVIEW

Within this chapter we've done literature review to develop an understanding of existing research and to do critical analysis on a particular topic. It helps to gain our knowledge and present our study in organized manner. In section 3.1 we have discussed research order. That includes research inquiries, purpose of doing literature work, keywords, literature review's criteria and PRISMA 2009 categorization. In section 3.2 we have done related work. And in section 3.3 we analyze the gaps after doing literature review.

3.1 Research order

Following are the steps to do literature review:

3.1.1 Research inquiries

Following are some research inquiries:

- How are lower and upper approximations calculated in DRSA?
- What are the methods for lowering the computational cost of DRSA?
- What is the value of conventional lower and upper approximation?
- How parallel computation affect calculation of DRSA approximation?
- How incremental approach affect DRSA approximation calculation?

3.1.2 Purpose of doing Literature Review

The aims of our literature review are as follows:

- To understand knowledge presented in existing research.
- To place our study in an historical perspective.
- To find out the gaps in the work that has already done.
- To provide the intellectual context for our own work.

- To gain knowledge and do critical analysis.
- To determine areas for future research.
- To effectively present our research and study.

3.1.3 Keywords

Keywords are listed down below:

- Dominance Based Rough Set Theory
- Rough Set Theory
- Lower approximation
- Upper approximation
- Computational complexity
- Execution time
- Memory consumption
- Complex structure
- Feature selection
- Classification

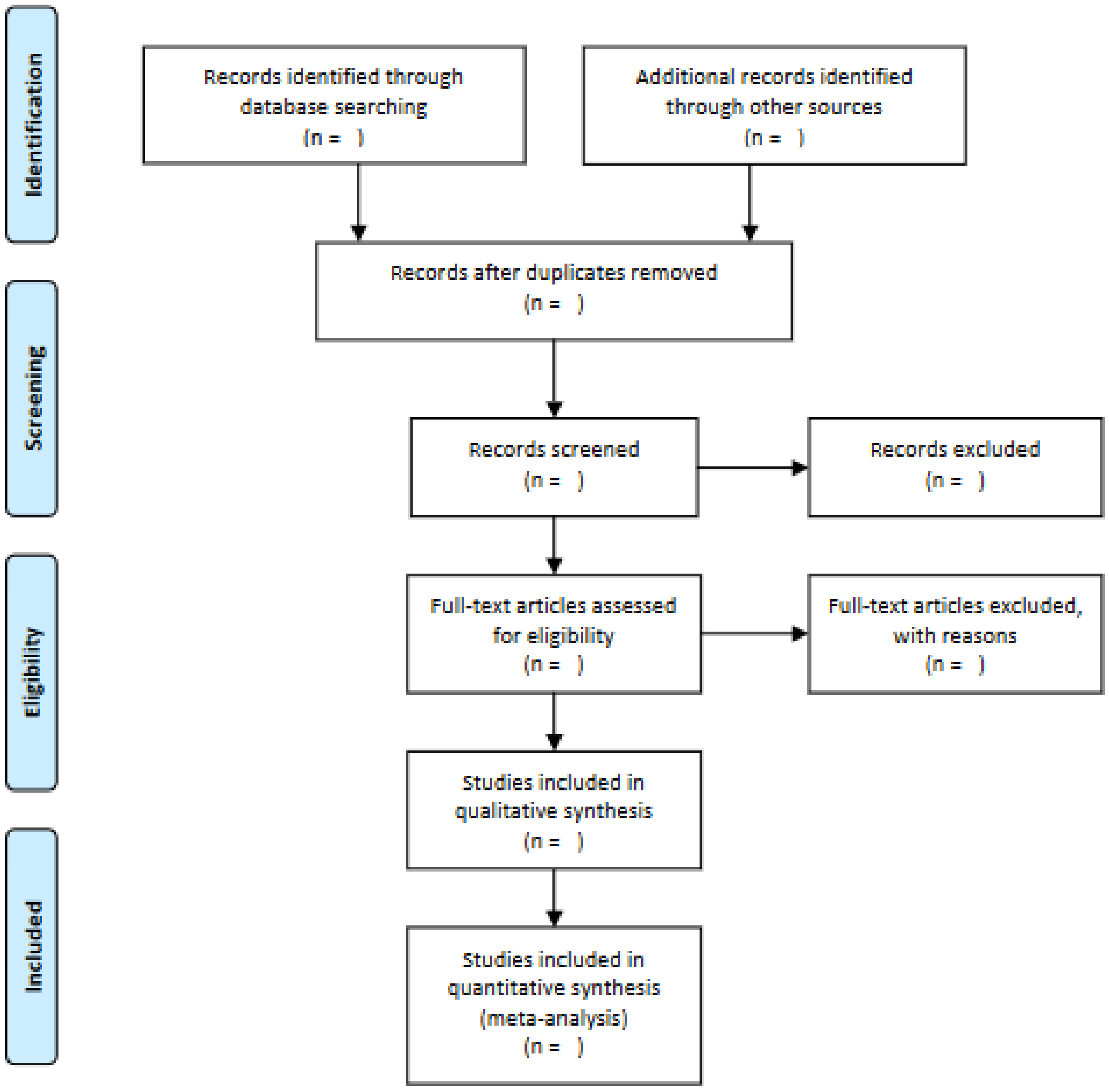
3.1.4 Literature Review's criteria

Literature review's selection and exclusion criteria is as follows:

- Popular databases like Science Direct, IEEE Xplore, etc. are targeted.
- Studies that are not relevant are omitted.
- Properly chosen paper related to keywords.
- Reputable and reliable researches are included.

3.1.5 Prisma 2009 Categorization

An explained breakdown of the search process is shown in figure 4.1-1. [48]



Identification
Screening
Eligibility
Included
Records identified through database searching (n =)
Additional records identified through other sources (n =)
Records after duplicates removed (n =)
Records screened (n =)
Records excluded (n =)
Full-text articles assessed for eligibility (n =)
Full-text articles excluded, with reasons (n =)
Studies included in qualitative synthesis (n =)
Studies included in quantitative synthesis (meta-analysis) (n =)

Figure 6: Prisma 2009 Categorization

3.2 Related Work

Dominance-Based Rough Set Approach (RST) and Rough Set Approach (DRSA) uses lower and upper approximations to explain data. The standard method of recalculating the upper and lower approximations of DRSA is not applicable in a dynamic environment where data is subject to constant changes. This increases structural complexity, execution time and memory consumption.

3.2.1 Conventional Approach

The first method is conventional method for calculating dominance related approximations. It is time consuming and expensive approach [28], [29], [30]. When dealing with a dynamic environment where data is always changing, DRSA estimates must be updated simultaneously. The computing time of the conventional DRSA approach increases with the quantity of the data since approximations have to be recalculated from beginning to finish for every modification.

3.2.2 Incremental Approach

The second method is to change the algorithm so that nothing needs to be recalculated as new data instances are added. We refer to this as an incremental approach.

Chen, Hongmei, et al. [31] focused on iterative methodology for updating VPRS approximations where dataset changes with time. It concentrated on information granulation and approximation for dynamic dataset where objects change over time. Significant decrease in calculation time in contrast to old method of calculation.

Cheng, Yi. et al. [32] offered two incremental ways to quickly construct crude fuzzy approximations where one depends on The additional ones on the dividing sets and the boundary set. They have compared this algorithm with non-incremental algorithm. Their proposed approach is time efficient.

Li, Shaoyong, Tianrui Li, and Dun Liu. et al. [33] In situations where data items change dynamically, an incremental technique for updating DRSA approximations was proposed. In dataset, when any object is added or removed, authors looked at the method that causes dynamic variation of DRSA approximations. It progressively updated the DRSA approximations in response to changes in the P-dominant sets and those items' P-dominated sets.

Luo, Chuan, Tianrui Li, and Hongmei Chen. [34] The authors state that dynamically maintaining approximations in set-valued structured decision systems is the main focus of the attribute generalization. Considering the main and dominated matrices in terms of dominance relation led to the development of a matrix-based method for calculating upward and downward approximations of decision classes. They came up with a step-by-step plan to enhance approximation computation that entails changing significant metrics without being aware of training datasets.

Luo, Chuan, et al. [35] presented two incremental algorithms for updating dynamic approximation maintenance of set-valued data, where values are related with an individual and change with time. Authors showed better results by using their proposed method on UCI datasets and artificial datasets.

Wang, Shu, et al. [36] focused on updating approximations for dynamic datasets where object values change over time. Author presented incremental approach to optimize DRSA approximations efficiently. They designed an algorithm that neglected unnecessary parameters to avoid redundant calculation. Author compares results with traditional method of DRSA. Computation time decreases for two-dimensional variation of objects and attributes.

Chen, Hongmei, Tianrui Li, and Da Ruan et al. [37] In the IODS, dynamically updated approximations of upward and downward unions were used to coarsen or refine attribute values. They have used incremental approach to reduce time taken by calculation of DRSA approximations. Proposed method is efficient and effective when compare with conventional method using UCI and empirical findings.

Liu, D., Li, T., Ruan, D. and Zou, W et al [38] suggested incremental paradigm, methodology, and algorithm for generating fascinating knowledge as the object collection changed over time.

Bouzayane, Sarra, and Ines Saad et al [39] suggested Incremental Periodic Prediction Multicriteria Approach (MAI2P). They developed a preference model that produces a set of choice rules by updating DRSA approximations incrementally. The results of experiments indicate that the most successful preference model is generated using a pessimistic cumulative technique, with accuracy and F-measure values of 0.89 and 0.66, respectively.

Table 2: Incremental approach related work

Paper	**Method and Technology**	**Improvement**	**Limitations**
[31]	DRSA based dynamic incremental approach	Improve complex computation	Hard to maintain scalability
[32]	DRSA based dynamic incremental approach	Significant decrease in computation time	Complex algorithm architecture
[33]	DRSA based dynamic incremental approach (matrix-based)	Upgrade approximation time	Proposed model is complicated
[34]	DRSA based dynamic incremental approach	Improve computation, efficient with execution time	Difficulty in algorithmic structure
[35]	DRSA based dynamic incremental approach	Provide better execution time to calculate approximations	Complex methodology
[36]	DRSA based dynamic incremental approach	Reduction in computation and execution time	Complicated architecture for algorithms
[37]	DRSA based dynamic incremental approach	For two-dimensional variations of objects and attributes, computation time reduces.	The construction of algorithm is complex
[38]	DRSA based dynamic incremental approach	Improve efficiency and effectiveness when compare with UCI and empirical findings	Difficult structure of proposed algorithm
[39]	DRSA based dynamic Multicriteria Approach for the Incremental Periodic Prediction (*MAI2P*)	Improved accuracy, advance applications	Structure of proposed model is complicated

3.2.3 Parallel Approach

The third approach involves partitioning the traditional algorithm and executing it in parallel across many hardware processors. This will expedite the process of calculations. It is also an excellent approach to speed up calculation, but the hardware becomes more expensive.

In [40] The authors suggested using a parallel technique to calculate estimates for a number of granule composition and breakdown procedures. Trials in a multi-core setting demonstrated a decrease in the amount of time needed to make DRSA-based judgements.

In [41] authors introduced parallel and incremental approximation calculation (PIAC) for lowering the cost of computing lower and upper approximations. Where they traversed the dataset parallelly and calculated approximations. Calculated approximations more efficiently when compare with traditional method.

In [42] authors computed dominance based rough set approach approximations in parallel. whereby all pointless computations from the traditional method are omitted, and parallel threads are used to speed up computing. The dominance relation calculation and comparison object repetition are ignored in the suggested method. When compared to prior methodologies, the study demonstrates improved performance in terms of time, cost, and memory.

In [43] for computing approximations in DRSA, the authors suggested a matrix-based method; they developed the relevant parallel methods on the graphics processing unit (GPU). A numerical example is provided to illustrate the practicality of the matrix-based method. Experimental investigations show that the parallel method performs much better than the old technique in terms of storage space and time consumption.

In [44] authors presented parallel approach to optimize effective computation of approximations that are vital in improving performance of data mining and related tasks. Proposed parallel approach is based on MapReduce Technique to deal with massive data. Results compared with the conventional approach. MapReduce parallel technique shows effective results for data mining.

In [45] authors have used parallel approach to compute approximations effectively and efficiently, that will help in reducing the time of decision making. Compared Parallel approach with traditional one made multiple process elements run at a time to save time and computational cost.

In [46] authors suggested a hierarchical attribute reduction technique. Hadoop MapReduce is used to make the algorithm work in parallel.

Table 3: Parallel approach related work

Paper	**Method and technology**	**Improvements**	**Limitations**
[40]	DRSA based dynamic parallel approach (Multicore environment)	Reduce execution time under multicore environment	Multicore environment, use more hardware components
[41]	DRSA based Parallel Incremental Approximation Calculation (PIAC)	Calculate approximations more efficiently	Multiprocessor, use more hardware components, expensive

[42]	DRSA based dynamic parallel approach	Show better performance with respect to time, cost and memory as compare to older methodology.	Use more hardware components, cost nonefficient
[43]	GPU-based matrix-based parallel methodology	Reduced intricacy of calculation.	Employ additional hardware parts
[44]	(MapReduce) Parallel approach	reduced computational complexity, more effective, and able to handle large amounts of data in cloud computing	Multiprocessor, use more hardware components, expensive
[45]	DRSA based dynamic parallel approach	Show effective results for data mining	Use multiprocessor, Expensive
[46]	Hadoop (MapReduce) method of Parallel hierarchical	Less computational complexity	Employ additional hardware parts (multiprocessor)

3.3 Analysis

After critically analyze the literature review, we have found gaps discussed below:

- Complex architecture design
- Use of more hardware

We have addressed both of the aforementioned gaps in our proposed strategy, which is covered in chapter 5. Our suggested method uses a straightforward algorithm that doesn't require any additional hardware to operate.

3.4 Gap Analysis

In one way or another, the aforementioned methods are all based on the conventional method for determining DRSA approximations. As we covered in Section III, some writers have proposed parallel processing, while other scholars have concentrated on an incremental approach that integrates the idea of dominance relation to carry out their study.

As was shown in Section II, there is a significant loss in algorithm performance when approximation sets are calculated using the conventional way. This is because computing rough approximations requires three computationally demanding procedures. To address the aforementioned problems, we have suggested a method based on the KD tree [49] in this study. Our suggested method is producing lower resource usage, including memory and time.

CHAPTER 4 : CHALLENGES IN COMPUTATION

In this chapter, we will first talk about the DRSA algorithm pseudo code. We then run an example of that algorithm in section 4.2. The final section goes into more detail about some elements that raise the computational cost.

4.1 Algorithm

Here, we'll go over the algorithm used to determine lower and upper approximations. This algorithm basically consists of four (4) parts. The following are these four (4) parts:

- Lower Approximation for $Cl_t^{\leq}$ $(\underline{P}(Cl_t^{\leq}))$
- Upper Approximation for $Cl_t^{\leq}$ $(\bar{P}(Cl_t^{\leq}))$
- Lower Approximation for $Cl_t^{\geq}$ $(\underline{P}(Cl_t^{\geq}))$
- Upper Approximation for $Cl_t^{\geq}$ $(\bar{P}(Cl_t^{\geq}))$

The lower approximation algorithm will first be explained. Next, we'll go over the upper approximation algorithm.

4.1.1 Lower Approximation

In order to calculate a lower approximation, there are three steps.

Step 1:

Class unions are discovered in the first step. The decision attribute in the dataset is traversed using a loop starting at 1 and ending at |U|. The specific class label is compared to each class label. The following pseudo-code represents step 1 for $Cl_t^{\geq}$.

```
For x=1 to |U|
  If $Cl_x \geq Cl_t$
    $Cl_t^{\geqq} = Cl_t^{\geqq} \cup Cl_x$
  End If
End For
```

For $Cl_t^{\leqq}$ pseudo-code is as shown below.

```
For x=1 to |U|
  If $Cl_x \leq Cl_t$
    $Cl_t^{\leqq} = Cl_t^{\leqq} \cup Cl_x$
  End If
End For
```

Step 2:

Two loops are needed for the $Cl_t^{\geqq}$ in the second step. Since we need to compute Dp^+ for each class union, the first loop goes from 1 to $Cl_t^{\geqq}$. To compare each object in the class union with every other record in the dataset, a second loop is run from 1 to |U|. The following pseudo-code for step 2 is for $Cl_t^{\geqq}$.

```
For x=1 to $|Cl_t^{\geqq}|$
  For y=1 to |U|
    If $X_y \geq X_{xt}$
      $Dp^+(X_x) = Dp^+(X_x) \cup X_y$
    End if
  End For
End For
```

Two loops are needed for the $Cl_t^{\leqq}$ in the second step. Since we need to compute Dp^{-} for each class union, the first loop goes from 1 to $Cl_t^{\leqq}$. To compare each object in the class union with every other record in the dataset, a second loop is run from 1 to |U|. The following pseudo-code for step 2 is for $Cl_t^{\leqq}$.

```
For x=1 to $|Cl_t^{\leqq}|$
  For y=1 to |U|
    If $X_y \leq X_{xt}$
      $Dp^{-}(X_x) = Dp^{-}(X_x) \cup X_y$
    End if
  End For
End For
```

Step 3:

In the third step, lower approximation is calculated using the conventional approach after Dp^{+} has been determined. We analyze how well the subset property holds between arrays. Three loops are required for this step. First and second loops are from 1 to $Cl_t^{\geqq}$ and Dp^{+}, respectively. To access the indexes of Dp^{+}'s two-dimensional arrays, use these two loops. Then, in order to determine whether the proper subset property is still valid, we must compare these indexes with the class unions. The third loop from 1 to the cardinality of the class union is used for this. The following pseudo-code for step 3 is for $Cl_t^{\geqq}$.

```
For x=1 to $|Cl_t^{\geq}|$
 For y=1 to $|Dp^{+}(X_x)|$
  For z=1 to $|Cl_t^{\geq}|$
     Calculate $Dp^{+}(X_{xj}) \subseteq Cl_{zt}^{\geq}$
   End For
 End For
End For
```

In the third step, lower approximation is calculated using the conventional approach after Dp^{-} has been determined. We analyze how well the subset property holds between arrays. Three loops are required for this step. First and second loops are from 1 to $Cl_t^{\leq}$ and Dp^{-}, respectively. To access the indexes of Dp^{-}'s two-dimensional arrays, use these two loops. Then, in order to determine whether the proper subset property is still valid, we must compare these indexes with the class unions. The third loop from 1 to the cardinality of the class union is used for this. The following pseudo-code for step 3 is for $Cl_t^{\leq}$.

```
For x=1 to $|Cl_t^{\leq}|$
 For y=1 to $|Dp^{-}(X_x)|$
  For z=1 to $|Cl_t^{\leq}|$
     Calculate $Dp^{-}(X_{xj}) \subseteq Cl_{zt}^{\leq}$
   End For
 End For
End For
```

4.1.2 Upper Approximation

In order to calculate an upper approximation, there are three steps.

Step 1:

Class unions are discovered in the first step. The decision attribute in the dataset is traversed using a loop starting at 1 and ending at |U|. The specific class label is compared to each class label. The following pseudo-code represents step 1 for $Cl_t^{\geqq}$.

```
For x=1 to |U|
If $Cl_x \geq Cl_t$
$Cl_t^{\geqq}$= $Cl_t^{\geqq}$ ∪ $Cl_x$
End If
End For
```

Step 2:

Two loops are needed for the $Cl_t^{\geqq}$ in the second step. Since we need to compute Dp^- against each class union, the first loop goes from 1 to $Cl_t^{\geqq}$. To compare each object in the class union with every other record in the dataset, a second loop is run from 1 to |U|. The following pseudo-code for step 2 is for $Cl_t^{\geqq}$.

```
For x=1 to $|Cl_t^{\geqq}|$
  For y=1 to |U|
    If $X_y \geq X_{xt}$
      $Dp^-(X_x) = Dp^-(X_x) \cup X_y$
    End if
  End For
End For
```

Two loops are needed for the $Cl_t^{\leq}$ in the second step. Since we need to calculate Dp^{+} against each class union, the first loop goes from 1 to $Cl_t^{\leq}$. To compare each instance in the class union with every other record in the dataset, a second loop is run from 1 to |U|. The following pseudo-code for step 2 is for $Cl_t^{\leq}$.

```
For x=1 to $|Cl_t^{\leq}|$
  For y=1 to |U|
    If $X_y \leq X_{xt}$
      $Dp^{+}(X_x) = Dp^{+}(X_x) \cup X_y$
    End if
  End For
End For
```

Step 3:

In the third step, lower approximation is calculated using the conventional approach after Dp^{-} has been determined. We analyze how well the subset property holds between arrays. Three loops are required for this step. First and second loops are from 1 to $Cl_t^{\geq}$ and Dp^{-}, respectively. To access the indexes of Dp^{-}'s two-dimensional arrays, use these two loops. Then, in order to determine whether the proper subset property is still valid, we must compare these indexes with the class unions. The third loop from 1 to the cardinality of the class union is used for this. The following pseudo-code for step 3 is for $Cl_t^{\geq}$.

```
For x=1 to $|Cl_t^{\geq}|$
 For y=1 to $|Dp^{-}(X_x)|$
  For z=1 to $|Cl_t^{\geq}|$
    Calculate $Dp^{-}(X_{xj}) \subseteq Cl_{zt}^{\geq}$
   End For
 End For
End For
```

In the third step, lower approximation is calculated using the conventional approach after Dp^{+} has been determined. We analyze how well the subset property holds between arrays. Three loops are required for this step. First and second loops are from 1 to $Cl_t^{\leq}$ and Dp^{+}, respectively. To access the indexes of Dp^{+}'s two-dimensional arrays, use these two loops. Then, in order to determine whether the proper subset property is still valid, we must compare these indexes with the class unions. For this, the final loop from 1 to the class union's relationship is employed. The following pseudo-code for step 3 is for $Cl_t^{\leq}$.

```
For x=1 to $|Cl_t^{\leq}|$
 For y=1 to $|Dp^{+}(X_x)|$
  For z=1 to $|Cl_t^{\leq}|$
    Calculate $Dp^{+}(X_{xj}) \subseteq Cl_{zt}^{\leq}$
   End For
 End For
End For
```

4.2 Example

To check the applicability of algorithm, we worked with an example ‘Evaluation for high school’ given in table 4 below.

Table 4: Evaluation of high school

Students	Mathematics	English	Computer	Score
X1	1	2	2	2
X2	3	3	1	3
X3	3	2	2	2
X4	2	2	3	3
X5	3	2	3	3
X6	2	2	1	1
X7	3	2	1	2
X8	2	3	1	2
X9	1	2	3	1
X10	1	1	2	1

In table 4 Students are objects called universe (U) represented by (X). There is total 10 objects. There are three conditional attributes (C) i.e., Mathematics, English and Computer. And one decision attribute (D) is represented by called score. In this example we choose preference order t=2.

4.2.1 Application of algorithm

In this section we applied our DRSA algorithm on this example given in table 4. We calculated lower and upper approximation sets.

For that we found class unions. i.e. upper-class union and downward-class union.

$$Cl_t^{\geq} = \{x1, x2, x3, x4, x5, x7, x8\}$$

$$Cl_t^{\leq} = \{x1, x3, x6, x7, x8, x9, x10\}$$

4.2.1.1 Lower Approximation

To calculate lower approximation, we followed 3 steps.

Step 1:

In this step, we calculated the objects belongs to $(Cl_t^{\geq})$ union class. In our example we have calculated $\underline{P}(Cl_t^{\geq})\ for\ t = 2$.

$Cl_t^{\geq}(x) = \{x1, x2, x3, x4, x5, x7, x8\}$

Step 2:

In this step, we have found Dp^{+} for each object identified by $Cl_t^{\geq}(x)$.

$For\ X1: Dp^{+}(X1) = \{X1, X3, X4, X5, X9, X10\}$

$For\ X2: Dp^{+}(X2) = \{X2, X3\}$

$For\ X3: Dp^{+}(X3) = \{X3, X5\}$

$For\ X4: Dp^{+}(X4) = \{X4, X5\}$

$For\ X5: Dp^{+}(X5) = \{X5\}$

$For\ X7: Dp^{+}(X7) = \{X3, X5, X7\}$

$For\ X8: Dp^{+}(X8) = \{X2, X8\}$

Step 3:

In this stage, we must compute the lower approximation result using the correct subset property. Each Dp^{+} is matched with a $\underline{P}(Cl_t^{\geq})$, and the appropriate subset attribute is determined. The graphic below depicts the appropriate subsets.

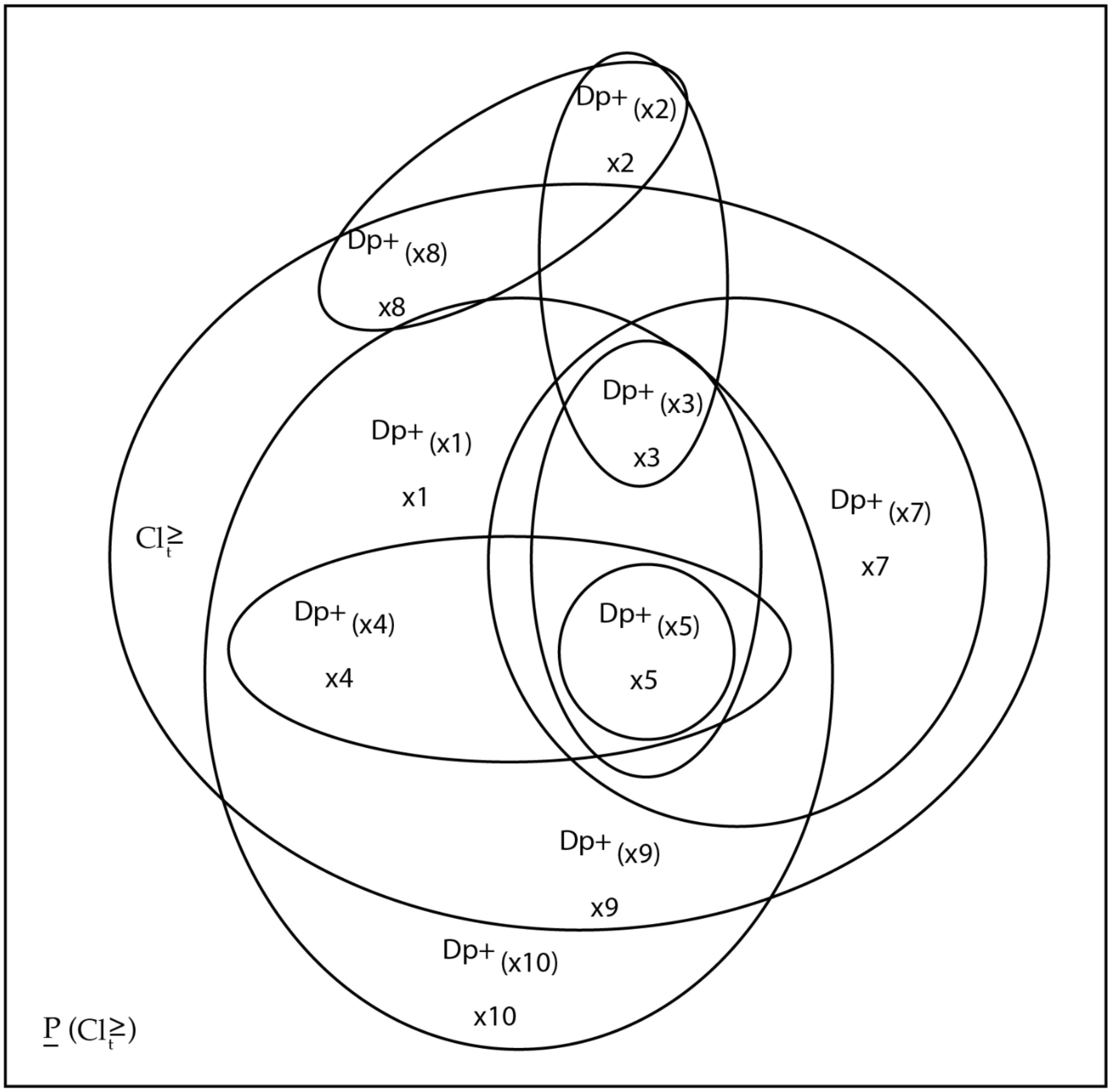


Figure 7: Subset property check for $\underline{P}(Cl_t^{\geq})$

Repeat above mentioned three steps to get $\underline{\boldsymbol{P}}(\boldsymbol{Cl_t^{\leq}})$

Step 1:

In this step, we calculated the objects belongs to $(Cl_t^{\leq})$ union class. In our example we have calculated $\underline{P}(Cl_t^{\leq})\ for\ t = 2$.

$Cl_t^{\leq}(x) = \{x1, x3, x6, x7, x8, x9, x10\}$

Step 2:

Calculate Dp^{-} for each individual object found in step 1

$For\ X1 :\ Dp^{-}(X1) = \{X1, X10\}$

$For\ X3 :\ Dp^{-}(X2) = \{X1, X3, X6, X7, X10\}$

$For\ X6 :\ Dp^{-}(X6) = \{X6\}$

$For\ X7 :\ Dp^{-}(X7) = \{X6, X7\}$

$For\ X8 :\ Dp^{-}(X8) = \{X6, X8\}$

$For\ X9 :\ Dp^{-}(X9) = \{X1, X9\}$

$For\ X10 :\ Dp^{-}(X10) = \{X10\}$

Step 3: This step gave us lower approximation subset. Shown in figure below.

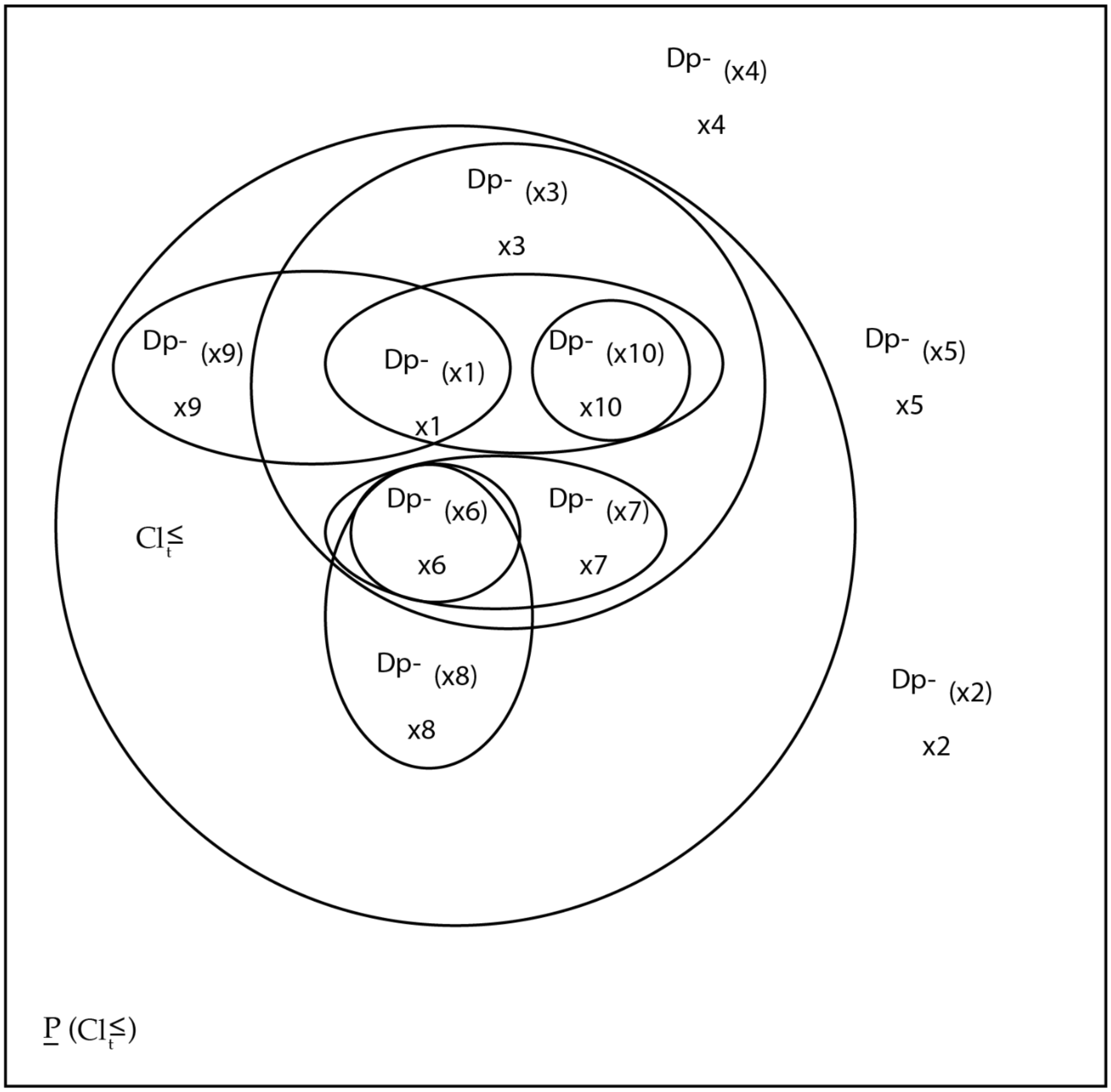


Figure 8: Subset property check for $\underline{P}(Cl_t^{\leq})$

4.2.1.2 Upper Approximation

Just like lower approximation, in upper approximation we followed 3 steps.

Step 1: We identify the class union in this step, exactly as we did in step 1 of the lower approximation. We estimated that the objects belong to the $Cl_t^{\geq}$ union class once more. In our case, we computed

$Cl_t^{\geq} = \{x1, x2, x3, x4, x5, x7, x8\}$

Step 2: we found D_p^{-} $for\ Cl_t^{\geq}$

$For\ X1 :\ Dp^{-}(X1) = \{X1, X10\}$

$For\ X2 :\ Dp^{-}(X2) = \{X2, X6, X7, X8\}$

$For\ X3 :\ Dp^{-}(X3) = \{X1, X3, X6, X7, X10\}$

$For\ X4 :\ Dp^{-}(X4) = \{X1, X4, X6, X9, X10\}$

$For\ X5 :\ Dp^{-}(X5) = \{X1, X3, X4, X5, X6, X7, X9, X10\}$

$For\ X7 :\ Dp^{-}(X7) = \{X6, X7\}$

$For\ X8 :\ Dp^{-}(X8) = \{X6, X8\}$

Step 3:

In this step intersection property was find using figure given below.

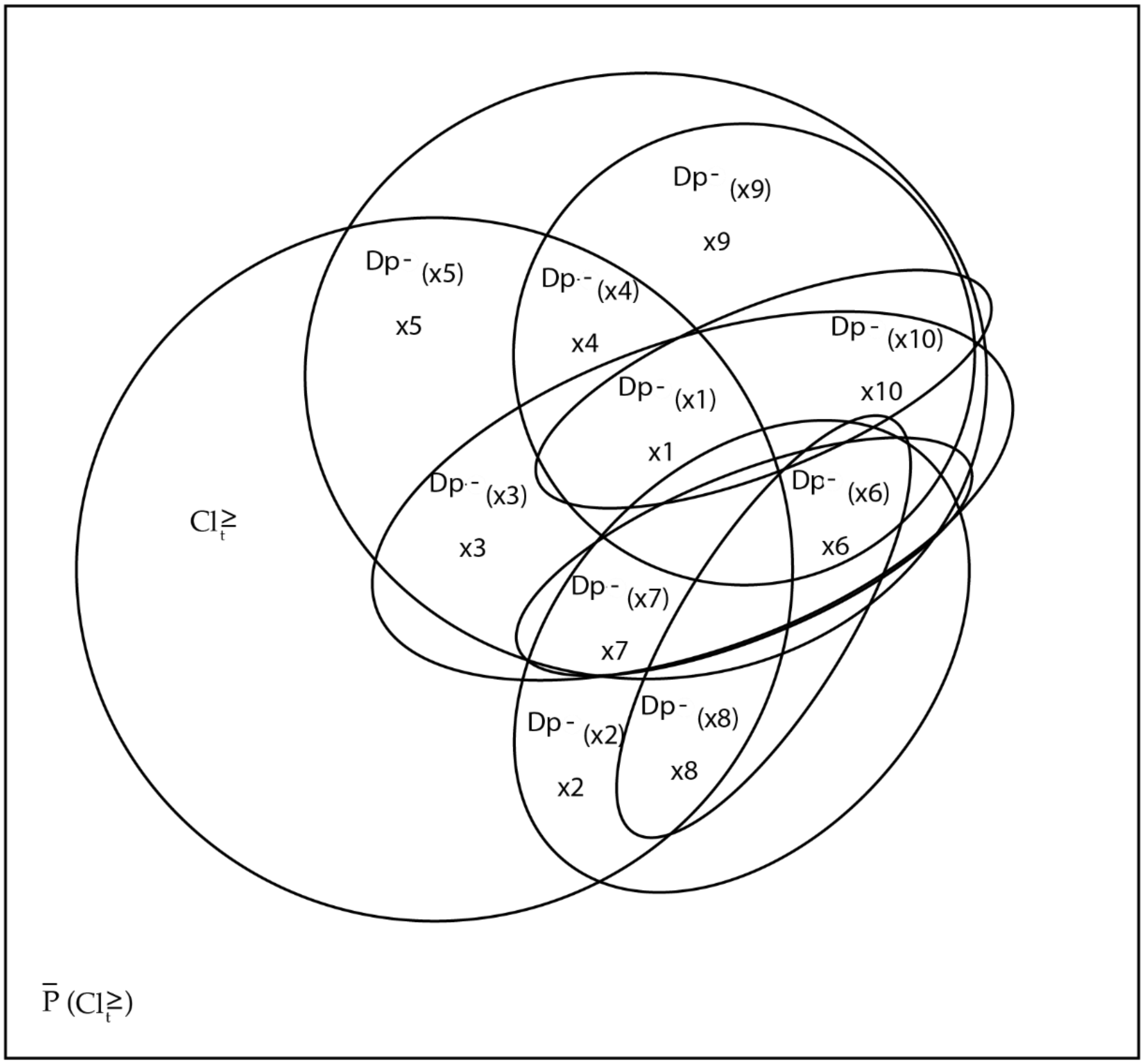


Figure 9: Intersection property check for $\overline{P}(Cl_t^{\geq})$

Repeat above mentioned three steps to get $\overline{\boldsymbol{P}}(\boldsymbol{Cl_t^{\leq}})$

Step 1:

$Cl_t^{\leq} = \{x1, x3, x6, x7, x8, x9, x10\}$

Step 2:

$For\ X1:\ Dp^{+}(X1) = \{X1, X3, X4, X5, X9, X10\}$

$For\ X3:\ Dp^{+}(X3) = \{X2, X5\}$

$For\ X6:\ Dp^{+}(X6) = \{X2, X3, X4, X5, X6, X7, X8\}$

$For\ X7:\ Dp^{+}(X7) = \{X3, X5, X7\}$

$For\ X8:\ Dp^{+}(X8) = \{X2, X8\}$

$For\ X9:\ Dp^{+}(X9) = \{X1, X4, X5, X9\}$

$For\ X10:\ Dp^{+}(X10) = \{X1, X3, X4, X5, X9, X10\}$

Step 3:

In this step intersection property was find using figure given below.

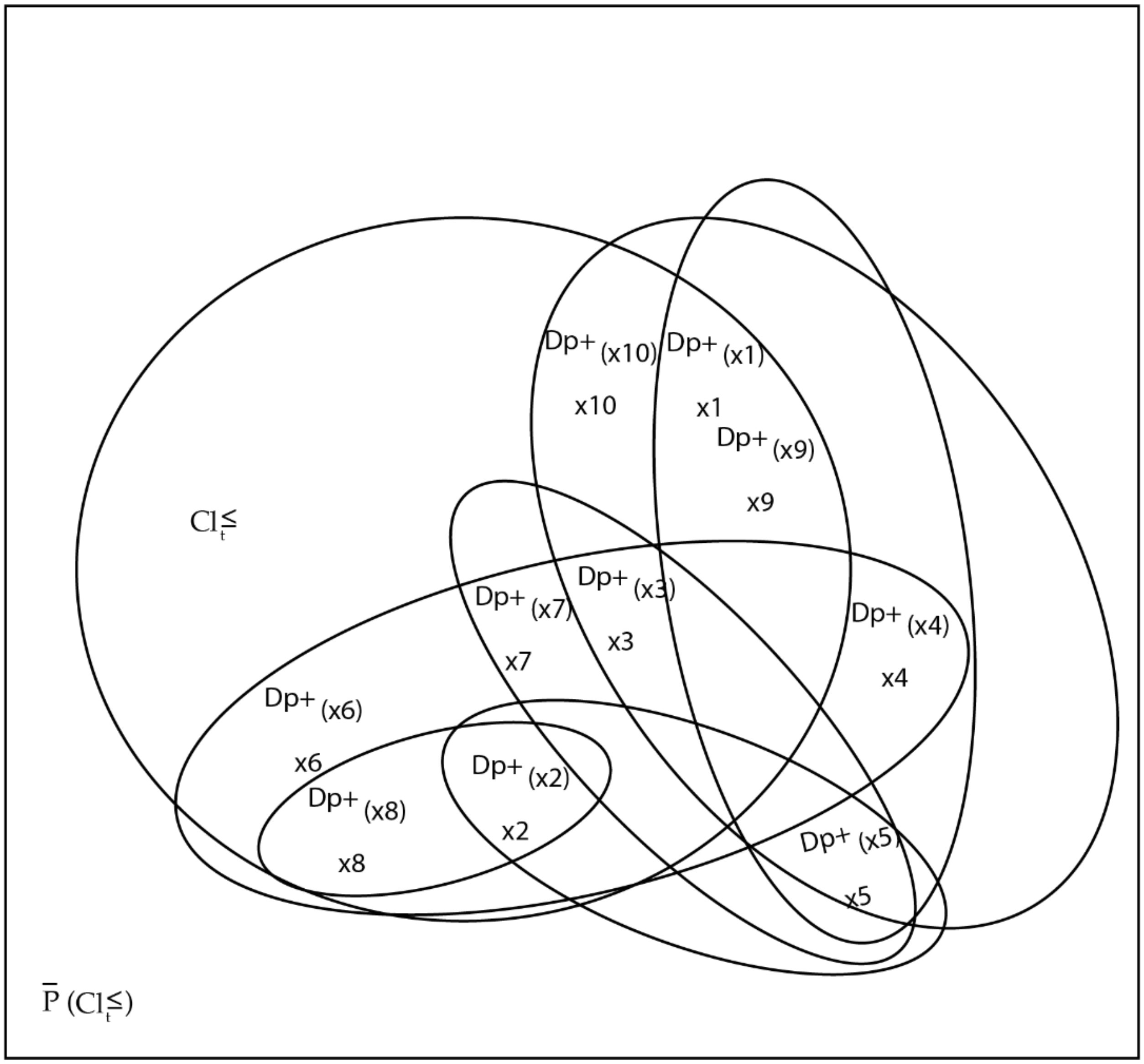


Figure 10: Intersection property check for $\overline{P}(Cl_t^{\leq})$

4.3 Causes of the rise in computational expenses

Calculating lower and upper approximations, as mentioned in section 4.1, needs three stages. Class unions are discovered in the first stage. To explore the decision attribute in the dataset, a loop from 1 to |U| is employed. Because there are three labels in the section 4.2 example, we must repeat this procedure eight (8) times. We need to compute $Cl_t^{\geq}$ and $Cl_t^{\leq}$ which are the merger of classes, both above and downward.

The example given above demonstrates that we must calculate Dp^{+} and $Dp^{-}$for each object, which is time consuming and computationally very expensive. If we talk about large dataset or the dataset that changes with time, calculation Dp^{+} and Dp^{-} of each object by iterating for the whole dataset consumes a lot of time and memory consumption. That will increase computational cost significantly.

CHAPTER 5 : PROPOSED METHODOLOGY

We have solved the problems with complicated algorithm structure and time-consuming complexity in our technique. The previously described issues were resolved by our suggested methods. We eschew every pointless step in the traditional DRSA method. and use our suggested approach to shorten the processing time.

Based on preference order, we have created two types of class unions. This categorization is based on a preferred order, which helps in data organization. Splitting the dataset into these two types makes it easier to analyze and handle the data depending on its unique properties as mentioned below in Table 5.

In this section, we'll talk about how to keep DRSA approximation sets up to date as an object set changes over time. We suppose that the previous approximation values are known. When new data is entered into the system, it is validated. This stage involves determining whether or not the data already exists in the old dataset. If the data exists, it will not be updated, and vice versa. This validation procedure guarantees that no duplicate data is handled, therefore conserving computing resources and avoiding unnecessary redundancy.

5.1 An algorithm to update Approximation Upon Insertion of new Object

In this section we considered an algorithm to update approximation upon insertion of new object. Table 5 gives the representation of symbols when new data enter into the system with time t+1. In which we considered two columns named as notions of DRSA at time ‘t’. and the second column represents notion of DRSA at time ‘t+1’, inserting of new object dynamically. Characterization of DRSA notions different symbols of decision table, upper approximation, downward approximation at time t and t+1.

Table 5: Details of symbols

Characterization	**Concepts of DRSA at time t**	**Concepts of DRSA at time t+1**
Decision-making system	U	U^{+}
Decision-making class t	Cl_t	Cl_t^{*}
More preferable decision class than 't'	$Cl_t^{\geq}$	$Cl_t^{\geq^*}$
Less preferable decision class than 't'	$Cl_t^{\leq}$	$Cl_t^{\leq^*}$
Lower Approximation for $\boldsymbol{Cl_t^{\leq}}$	$\underline{P}(Cl_t^{\leq})$	$\underline{P}^{*}(Cl_t^{\leq})$
Upper Approximation for $\boldsymbol{Cl_t^{\leq}}$	$\bar{P}(Cl_t^{\leq})$	$\bar{P}^{*}(Cl_t^{\leq})$
Lower Approximation for $\boldsymbol{Cl_t^{\geq}}$	$\underline{P}(Cl_t^{\geq})$	$\underline{P}^{*}(Cl_t^{\geq})$
Upper Approximation for $\boldsymbol{Cl_t^{\geq}}$	$\bar{P}(Cl_t^{\geq})$	$\bar{P}^{*}(Cl_t^{\geq})$

Table 6: Pseudocode to set union of classed using preference order

```
• Input: $U' = U^{+} = U \cup U'$
        ▪ t= Total Number of Classes
• Output: $Cl_t^{*}$ , $Cl_t^{\geq^*}$, $Cl_t^{\leq^*}$
• For y=1 to t
    o For i=1 to $|U^{+}|$
        ▪ IF ($x_i$ . t) ∈ Class y
        ▪ $Cl_t^{*} = Cl_t^{*} \cup \{x_i\}$
        ▪ Else IF ($x_i$ . t) ∈ more preference class than class y
        ▪ $Cl_t^{\geq^*} = Cl_t^{\geq^*} \cup \{x_i\}$
        ▪ Else IF ($x_i$ . t) ∈ less preference class than class y
                  $Cl_t^{\leq^*} = Cl_t^{\leq^*} \cup \{x_i\}$
        ▪ End IF
    o End For
End For
```

The following next step is to implement a KD (K-Dimensional) tree approach. This method is used to determine lower and upper approximations. A KD tree is a data structure that organizes points in a multidimensional space, making search and retrieval operations more efficient. The methodology uses the KD tree approach to optimize the overall computing efficiency of the operation and reduce execution time. The KD tree technique was used to minimize the complexity and execution time of the DRSA (Dominance-based Rough Set technique) algorithm. The approach attempts to increase the efficiency of the data analysis process by utilizing the KD tree, making it more effective and time-saving.

Table 7: Pseudocode of KD tree

```
• Input:    start-node,   //initial point
• Output:  kd,     //representation of kd tree
• Pre:       None
• Post:      start-node= node-rep(kd)^ |s-legal-kdtree(kd)|
    i. IF start-node is empty THEN return the empty kd-tree
   ii. Call pivot choosing procedure, which returns two values,
          n: = a member of node
          split: = the splitting dimensions
  iii. d: = domain vector of n
   iv. Node’: = node with n removed
    v. r: = range vector of n
   vi. nodeleft: ={(d’,r’) ∈ node’ | d’split ≤ dsplit}
  vii. noderight: ={(d’,r’) ∈ node’ | d’split ≥ dsplit}
 viii. kdleft: =recursively construct kd tree from nodeleft
   ix. kdright: =recursively construct kd tree from noderight
    x. kd: = <d, r, split, kdleft, kdright>
Proof: By inducing on the length of start-node and the definitions of node-rep and s-legal-kdtree
```

Table 7 represents the algorithm of KD tree working. A KD tree, also known as a data structure called a "k-dimensional tree" is used to arrange points in a k-dimensional space. It is very handy for performing effective closest neighbor searches. Here's a more in-depth description of how a KD tree works:

Construction

The first step in constructing a KD tree is to choose a splitting axis [50]. This axis is chosen depending on a variety of factors, such the dimension with the largest variance or the one that splits the points evenly. After selecting an axis, the points are split into two categories according on where they are on that axis.

Recursive Subdivision

For each subset, the dividing procedure is continued recursively until a termination condition is fulfilled. This constraint might be a maximum depth limit, a minimum number of points in a leaf node, or any other requirement that the implementation defines. Balancing: It is critical to balance the tree in order to provide efficient search operations. This can be accomplished by choosing a suitable splitting axis at each level of the tree, or by employing approaches such as median splitting or randomness. When executing a closest neighbor search in a KD tree [51], the algorithm starts at the root node and traverses the tree recursively depending on the splitting criteria. It chooses which child node to visit at each level depending on the query point's position relative to the splitting plane. This procedure is repeated until a leaf node is reached.

Backtracking

When the algorithm reaches a leaf node, it goes back to the parent nodes to see if there are any closer locations in the other subtree. The distance between the question points and the dividing plane is calculated. If the distance is less than the current best distance, the algorithm moves on to the next subtree. Pruning: The method may encounter nodes that can be pruned throughout the backtracking phase if their bounding box or distance to the query location is greater than the current best distance. This aids in reducing needless computations and increasing search efficiency. Termination: The search ends when all viable nodes have been visited or when a specified condition, such as identifying the precise nearest neighbor or exceeding a predetermined distance threshold, is fulfilled.

In the next step, we need to update lower and upper approximations by using above mentioned KD tree rules. We carried out our computation task using the KD tree algorithm. The idea behind the algorithm is to use a binary tree structure to divide the data points into smaller regions. we were able to efficiently find nearest neighbors using this algorithm [52] as well as complete other calculations. For simplicity and usability, we implemented the algorithm, which has ten rules, in a single line of code named as kd_tree mentioned in above Table 7.

Table 8: Details of KD tree symbols

```
Input      kd_tree, t=2
Output     $Cl^*_{\leq}, Cl^*_{\geq}, Cl^*_t, Cl_t^{\geq^*}, Cl_t^{\leq^*}$
$\underline{P}^*(Cl_t^{\leq}), \overline{P}^*(Cl_t^{\leq}), \underline{P}^*(Cl_t^{\geq}), \overline{P}^*(Cl_t^{\geq})$
  ▪ Calculate $Cl^*{}_t^{\leq}$ and $Cl^*{}_t^{\geq}$
      • Pass the kd_tree object, the value of t, and the count_only parameter set to True
        to the query_radius function of the kd_tree object
      • Store the result of the query_radius function in a variable named $Cl^*_{\leq}$ & $Cl^*{}_t^{\geq}$
  ▪ Calculate $\underline{P}^*(Cl_t^{\leq}), \overline{P}^*(Cl_t^{\leq}), \underline{P}^*(Cl_t^{\geq}), \overline{P}^*(Cl_t^{\geq})$
    1) Create Empty lists of $\underline{P}'(Cl_t^{\leq}), \overline{P}'(Cl_t^{\leq}), \underline{P}^*(Cl_t^{\geq}), \overline{P}^*(Cl_t^{\geq})$
    2) For i, row in iterrows for $Cl^*{}_t^{\leq}$
      • Set i = current point in the kd_tree
      • Reshape the current point into a 1D array and pass it to the query_radius
        function of the kd_tree
      • Set the radius= 0 and return_distance= False
```

```
        • Sort the results of the query_radius function in ascending order and store them in $Dp_{\leq}^{+}$
        • Repeat above two steps for $Dp_{\leq}^{-}$
        • Now set radius=t, and return_distance=False
        • Sort the results of the query_radius function in ascending order and store them in $Dp_{\leq}^{-}$
    3) $\underline{P}'(Cl_t^{\leq})$ = Flatten the $Dp_{\leq}^{+}$ array and add 1 to each element
    4) $\bar{P}'(Cl_t^{\leq})$ = Flatten the $Dp_{\leq}^{-}$ array and add 1 to each element
    5) Append $\underline{P}'(Cl_t^{\leq}), \bar{P}'(Cl_t^{\leq})$ into $\underline{P}^{*}(Cl_t^{\leq}), \bar{P}^{*}(Cl_t^{\leq})$ lists.
    6) Repeat whole process to calculate $\underline{P}^{*}(Cl_t^{\geq}), \bar{P}^{*}(Cl_t^{\geq})$ as well
    7) End For Loop
```

5.2 Flow Chart of proposed methodology

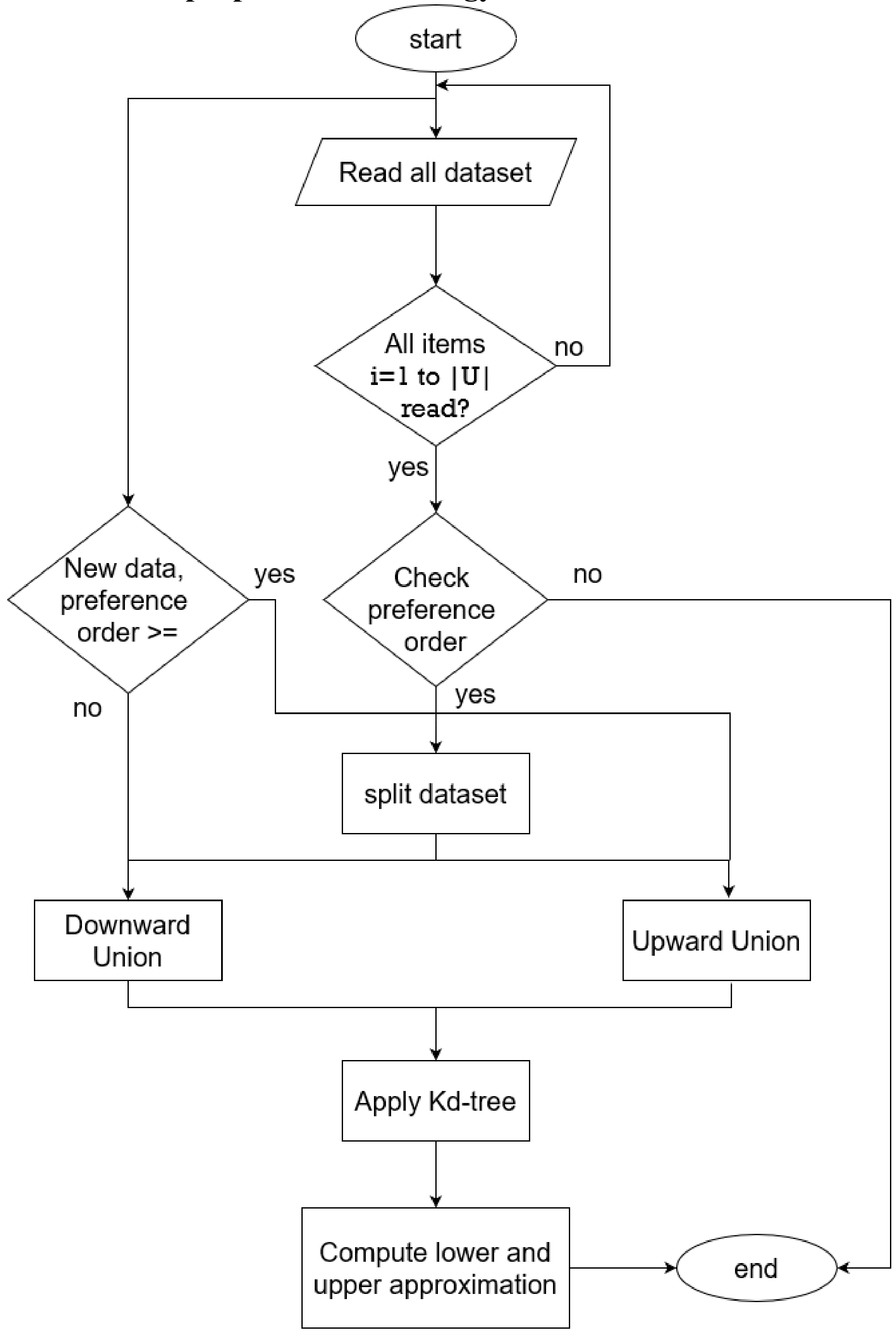


Figure 11: Flow chart of proposed methodology

5.3 Illustrative Example

We take into consideration the following sample information system:

Table 9: Sample decision system

Universe	**English**	**Mathematics**	**Grades**
X1	B	B	Good
X2	C	B	Good
X3	B	C	Good
X4	A	B	Very Good
X5	A	A	Excellent
X6	A	B	Very Good

Consider that we already know about the decision classes and approximations.

Whenever a new object is inserted, the approximations are updated.

We assumed that a new set of objects would be used to further clarify the implementation of our suggested strategy $U^{+} = \{\boldsymbol{X7}, \boldsymbol{X8}, \boldsymbol{X9}\}$ details of the conditional and decision attribute are provided in the decision system as

{Universe, English, Mathematics, Grade}:

- $\mathbf{(X7, C, B, Good)}$
- **(X8, B, A, Very Good)**
- **(X9, A, A, Excellent)**

Take new objects X7, X8, X9 into consideration. In order to update the approximation sets, we first update the decision classes and the corresponding higher/lower preference ordered classes.

After adding two new objects to the same decision system:

i. X7 belongs to les preference class and X8 and X9 belongs to preference class as per preference order t=2.

ii. More preference ordered class:

$Cl^*{}_t \geq = \{X4, X5, X6, X8, X9\}$

iii. Less preference ordered class:

$Cl^*{}_t \leq = \{X1, X2, X3, X4, X6, X7, X8\}$

iv. Updated Lower Approximation for

$Cl^*{}_t \geq$ is:

- $For\ X4 :\ Dp^{+}(X4) = \{X4, X5, X6, X9\}$
- $For\ X5 :\ Dp^{+}(X5) = \{X5, X9\}$
- $For\ X6 :\ Dp^{+}(X6) = \{X4, X5, X6, X9\}$
- $For\ X8 :\ Dp^{+}(X8) = \{X5, X8, X9\}$
- $For\ X9 :\ Dp^{+}(X9) = \{X5, X9\}$

v. Updated Lower Approximation for

$Cl^*{}_t \leq$ is:

- $For\ X1 :\ Dp^{-}(X1) = \{X1, X4, X5, X8, X9\}$
- $For\ X2 :\ Dp^{-}(X2) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$
- $For\ X3 :\ Dp^{-}(X3) = \{X3, X4, X5, X6, X8, X9\}$
- $For\ X4 :\ Dp^{-}(X4) = \{X4, X5, X6, X9\}$
- $For\ X6 :\ Dp^{-}(X6) = \{X4, X5, X6, X9\}$
- $For\ X7 :\ Dp^{-}(X7) = \{X1, X2, X4, X5, X6, X7, X8, X9\}$
- $For\ X8 :\ Dp^{-}(X8) = \{X5, X8, X9\}$

vi. Updated Upper Approximation for

$Cl^{*}{}_{t} \geq$ is:

- $For\ X4 :\ Dp^{+}(X4) = \{X1, X2, X3, X4, X6, X7\}$
- $For\ X5 :\ Dp^{+}(X5) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$
- $For\ X6 :\ Dp^{+}(X6) = \{X1, X2, X3, X4, X6, X7\}$
- $For\ X8 :\ Dp^{+}(X8) = \{X1, X2, X3, X7, X8\}$
- $For\ X9 :\ Dp^{+}(X9) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

vii. Updated Upper Approximation for

$Cl^{*}{}_{t} \leq$ is:

- $For\ X1 :\ Dp^{-}(X1) = \{X1, X2, X3, X7\}$
- $For\ X2 :\ Dp^{-}(X2) = \{X2, X7\}$
- $For\ X3 :\ Dp^{-}(X3) = \{X3\}$
- $For\ X6 :\ Dp^{-}(X6) = \{X1, X2, X3, X4, X6, X7\}$
- $For\ X8 :\ Dp^{-}(X8) = \{X1, X2, X3, X7, X8\}$
- $For\ X9 :\ Dp^{-}(X9) = \{X1, X2, X3, X4, X5, X6, X7, X8, X9\}$

CHAPTER 6 : RESULTS AND ANALYSIS

In this section, we examine the effectiveness of the suggested method for updating approximations as objects migrate and emigrate. In section 6.1 we have explained datasets. Section 6.2 includes characteristics of datasets. In section 6.3, we have done our experimental measures. Section 6.4 shows evaluation design. Section 6.5 includes results and discussions.

6.1 Dataset

We used ten publicly available datasets from the UCI library to evaluate the effectiveness of our proposed method [47].

- 1st dataset that we have used was breast cancer Coimbra. There are ten statistical variables and a binary dependent variable with a binary dependence that indicates whether breast cancer is present or not are present. Anthropometric data and factors collected during normal blood analysis serve as predictors. If these predictors are true, models of prediction based on them might be utilized as a biomarker for breast cancer. There are no missing values. There are 116 instances and having 10 features. Dataset is multivariant/integer type.

- The second dataset that we have chosen was caesarian section classification. This dataset provides information about the caesarian section results of 80 pregnant women who had the most common delivery difficulties in the medical industry. It has 80 instances and 5 features, having univariant/integer type.

- 3rd dataset was lung cancer. The information provided outlined three forms of problematic lung cancers. Having 32 instances and 56 features. The type of dataset is multivariant/integer.

- 4th dataset was glass. Criminological research served as the impetus for the study of glass classification since glass left at the scene of crime might be used as evidence. It has 214 instances and 9 features. Having no missing values. The type of dataset is multivariant/real.

- Fifth dataset chosen was Haberman. Having 306 instances and 3 features. The dataset comprises instances from research on the survival of breast cancer patients who had surgery. The type of dataset is multivariant/integer.

- 6th dataset that we have used was Iris. This is one of the oldest datasets used in the classification literature, and it is frequently used in statistics and machine learning. The data set is divided into three classes, each with 50 instances, and each class represents a different species of iris plant. One class may be separated linearly from the other two; the latter cannot be separated linearly from each other. It has 4 attributes. The type of iris dataset is real.

- 7th dataset was concrete compressive strength. The most significant material in civil engineering is concrete. The compressive strength of concrete is a highly nonlinear function of age and constituents. The dataset has 1.03k instances and 9 features. Dataset is of real type.

- Eighth dataset was Iranian churn dataset. This information was gathered at random from an Iranian telecom company's database during a 12-month period. A total of 3150 rows of data, one for each client, contain information for 13 columns. The characteristics included in this dataset. Call failures, SMS frequency, number of complaints, number of separate calls, subscription term, age group, fee amount, kind of service, seconds of usage, position, the amount of use, and Customer Value are all factors to consider. Except for attribute churn, all of the attributes are based on aggregated data from the first 9 months. The churn labels represent the customers' status at the end of a year. The minimum planning gap is three months. It has 3.15k instances and 13 features, having integer type.

- 9th dataset was waveform dataset generator version 1. The characteristic of dataset is multivariant/data generator. It has 5k instances and 21 features. The subject area of dataset physics and chemistry. Feature type is real.

- 10th dataset was letter recognition. The objective is to recognize each of the many black-and-white rectangular pixel representations as one of the 26 capital letters in the English alphabet. The character representations were created using 20 distinct typefaces, and each letter within these fonts was randomly deformed to create a file containing 20,000 unique stimuli. Each stimulus was transformed into 16 primitive numerical properties and scaled to fit within a range of integer values ranging from 0 to 15. In most cases, we train on the first 16000 items and then use the resultant model to predict the letter category for the other 4000. More information may be found in the aforementioned article. It has 20k instances and 64 features.

To thoroughly examine the effectiveness of our suggested algorithms, we compare them to the standard DRSA algorithm that operated in a dynamic environment. With 5% of the original dataset size as the variation ratio, we compared the suggested and standard techniques. With the understanding that the value set would remain the same, we chose random values for the conditional and decision characteristics of new objects. Using the new approach and the traditional DRSA algorithm, we updated the DRSA approximations for outdated approximation sets. By computing the percentage of execution time that was saved, we were able to compare the computational time of the two algorithms. When compared to the usual method, our suggested approach updated approximations in less time, with an average reduction of 99.2%.

6.2 Characteristics of dataset

We have tested our proposed algorithm on UCI ten datasets. Table 10 shows the characteristics of datasets including their instance (number of rows or objects) and attributes (number of columns or features) and type. Despite of their type, some datasets have more instance like concrete compressive strength, Iranian churn dataset, Waveform database generator version1 and Letter recognition. And rest of the datasets have a smaller number of instances. UCI datasets of different types like multivariant and univariant as well as integer and real numbers.

Table 10: Characteristics of UCI dataset

Dataset	Instance	Attributes	Type
Breast cancer Coimbra	116	10	Multivariate/Integer
Caesarian section classification	80	5	Univariate/Integer
Lung cancer	32	56	Multivariate/Integer
Glass	214	9	Multivariate/Real
Haberman	306	3	Multivariate/Integer
Iris	150	4	Real
Concrete compressive strength	1.03k	9	Real
Iranian churn dataset	3.15k	13	Integer
Waveform database generator version 1	5k	21	Multivariate, Data-Generator
Letter recognition	20k	64	Integer

6.3 Experimental measure

Below is a brief description of the experimental settings that were used to evaluate and execute the recommended and standard techniques.

6.3.1 KD Tree

A data structure called a KD tree, often referred to as a k-dimensional tree, is used to arrange points in a k-dimensional space. It is very handy for performing effective closest neighbor searches. The KD tree technique was used to minimize the complexity and execution time of the DRSA (Dominance-based Rough Set technique) algorithm. The approach attempts to increase the efficiency of the data analysis process by utilizing the KD tree, making it more effective and time-saving.

6.3.2 Execution Time

"Execution time" refers to the amount of time an algorithm must analyses data before generating an output. We evaluated the algorithm execution time using a system timer in order to evaluate the performance of the suggested technique to the conventional approach and other relevant contemporary techniques. We tested our method on ten UCI datasets and determined the average execution time. Equation 10 was utilized to calculate the percentage reduction in the duration of the suggested method.

$$\boldsymbol{Percentage\ decrease} = \left(\frac{\boldsymbol{T_1 - T_2}}{\boldsymbol{T_1}}\right) \times \mathbf{100} \tag{10}$$

where ‘T1’ denotes the traditional algorithm's execution time and ‘T2’ denotes the suggested algorithm's execution time.

Table 11: Hardware specifications

Hardware	**Specifications**
Processor	AMD Ryzen 5 3.60 GHz
RAM	32 GB
Generation	AMD Ryzen 6
Cache	512 Bytes

Table 11 shows hardware specifications. Processor that we have used for our research work was AMD Ryzen5 3.60 GHz, generation of AMD Ryzen 6. Our hardware has 32 GB RAM. Cache was about 512 bytes.

6.4 Evaluation design

We examined both methods (conventional approach and proposed approach) using the parameters listed below:

- Execution environment
- Algorithmic parameters

6.4.1 Execution environment

Table 11 lists the hardware and its specifications that was used to implement the proposed and conventional algorithms. AMD Ryzen (AMD Ryzen 5 3.60 GHz, RAM 32 GB) is used to execute both scripts on the ten datasets specified at the beginning of section 6. When comparing both methods, the execution environment is same. When the algorithms are executing, no further processes are active. During implementation, great effort is taken to ensure that the processor state remains consistent.

6.4.2 Algorithmic Parameters

Conventional and proposed algorithm takes following parameters for information system:

- IS represents information system
- 'U' represents universe or finite number of objects
- 'Q' is $(C \cup D)$ represents a finite number of conditional and decision attributes
- V = $U_{q \in Q} V_q$ where V_q is the set of values of criteria q
- f represents function of $f(x, q)$ which assigns a particular value V_q to an object x for attribute q.

6.5 Results and discussion

We updated the DRSA approximations for out-of-date approximation sets using the novel strategy and the conventional DRSA technique. We were able to compare the computational times of the two techniques by determining the percentage of time required for execution that was saved. With an average drop of 99.2%, our suggested process updated estimates faster than the traditional approach.

Table 12: Calculation time for the standard, parallel, and suggested approaches after adding new objects

Dataset	**Time Taken(s)**			**Percentage decrease in time (%)**
	Conventional Algorithm	**Parallel Algorithm**	**Suggested Algorithm**	
Breast Cancer Coimbra	6.75	2.37	0.59	91.2%
Caesarian Section Classification	3.71	29.88	0.56	84.9%
Glass	18.7	6.75	0.71	96.2%
Haberman	49.9	16.7	1.67	97%
Iris	11.4	3.98	0.5	95%
Concrete Compressive strength	309.2	108.2	2.1	99.3%
Iranian Churn Data	450.9	157.8	2.9	99.3%
Waveform database generator version1	489.1	171.8	3.1	99.3%
Letter Recognition	1545.2	541.8	11.2	99.3%
Lung Cancer	1.04	0.36	0.54	92.6%

Dominance based rough set theory (DRSA) takes more time to calculate lower and upper approximation because comparing dominance relations among data points can be complex. We have addressed the problems associated with complicated algorithmic structure and lengthy execution times in our methods. We do not do any traditional DRSA approach measures that are not essential. We have used KD tree algorithm to decrease computational time of DRSA approaches for calculating lower and upper approximations.

The KD tree, in the first place, arranges the data points in a hierarchical structure to facilitate effective search operations. The algorithm can quickly find important areas of the data space, which lowers the number of comparisons required. As a result, compared to linear search algorithms, search operations have a much lower complexity. Additionally, during the search process, the KD tree method reduces irrelevant sections. In Table 12, we have compared conventional, parallel and proposed algorithmic results applied on UCI ten datasets.

Our proposed algorithm shows reduce execution time on comparison. In conclusion, the hierarchical structure, effective search operations, and reducing mechanism of the KD tree algorithm give strong evidence for its ability to shorten the computing time of DRSA techniques when calculating lower and higher approximations. It is a useful tool for increasing the effectiveness of DRSA calculations since it can handle high-dimensional spaces and lowers number of comparisons.

Table 13: Execution time for conventional approach (upward and downward union class)

Dataset	**Traditional methodology time duration (s)**	
	Set of approximation for $Cl_t^{\leq}$	**Set of approximation for $Cl_t^{\geq}$**
Breast Cancer Coimbra	1.06	1.5
Caesarian Section Classification	20.01	19.8
Glass	4.1	3.8
Haberman	11.02	10.25
Iris	2.6	2.6
Concrete Compressive strength	68.57	70.6
Iranian Churn Data	102.5	110.1
Waveform database generator version 1	121.1	123.3

Letter Recognition	330.15	324.4
Lung Cancer	0.17	0.2

Table 13 shows Execution time for conventional approach (upward and downward union class) of 10 UCI datasets that are available online. For Breast Cancer Coimbra, execution time for $Cl_t^{\leqq}$ = 1.06 s and for $Cl_t^{\geqq} = 1.5s$.

For Caesarian Section Classification, execution time for $Cl_t^{\leqq}$ = 20.01 s and for $Cl_t^{\geqq} = 19.8s$. For Glass dataset, $Cl_t^{\leqq}$ = 4.1 s and for $Cl_t^{\geqq} = 3.8s$. Haberman has $Cl_t^{\leqq}$ = 11.02 s and for $Cl_t^{\geqq} = 10.25s$

For iris, execution for conventional approach is $Cl_t^{\leqq}$ = 2.6 s and for $Cl_t^{\geqq} = 2.6s$. Concrete compressive strength has $Cl_t^{\leqq}$ = 68.57 s and for $Cl_t^{\geqq} = 70.6s$. Iranian churn data has $Cl_t^{\leqq}$ = 102.5s and for $Cl_t^{\geqq} = 110.1s$. For waveform database generator version 1, $Cl_t^{\leqq}$ = 121.1 s and for $Cl_t^{\geqq} = 123.3s$. Letter recognition has $Cl_t^{\leqq}$ = 330.15s and for $Cl_t^{\geqq} = 324.4s$. For lung cancer $Cl_t^{\leqq}$ = 0.17s and for $Cl_t^{\geqq} = 0.2s$.

Table 14: Execution time for proposed approach (upward and downward union class)

Dataset	**Proposed methodology time duration (s)**	
	Set of approximation for $Cl_t^{\leqq}$	**Set of approximation for $Cl_t^{\geqq}$**
Breast Cancer Coimbra	0.025	0.025
Caesarian Section Classification	0.03	0.031
Glass	0.029	0.029
Haberman	0.168	0.16
Iris	0.02	0.023

Concrete Compressive strength	0.2	0.21
Iranian Churn Data	0.31	0.31
Waveform database generator version1	0.29	0.3
Letter Recognition	2.39	2.4
Lung Cancer	0.02	0.023

Table 14 shows Execution time for proposed approach (upward and downward union class) of 10 UCI datasets that are available online. For Breast Cancer Coimbra, execution time for $Cl_t^{\leqslant}$ = 0.025 s and for $Cl_t^{\geqslant} = 0.025s$.

For Caesarian Section Classification, execution time for $Cl_t^{\leqslant}$ = 0.03s and for $Cl_t^{\geqslant} = 0.031s$. For Glass dataset, $Cl_t^{\leqslant}$ = 0.029s and for $Cl_t^{\geqslant} = 0.029s$. Haberman has $Cl_t^{\leqslant}$ = 0.168s and for $Cl_t^{\geqslant} = 0.16s$

For iris, execution for conventional approach is $Cl_t^{\leqslant}$ = 0.02s and for $Cl_t^{\geqslant} = 0.023s$. Concrete compressive strength has $Cl_t^{\leqslant}$ = 0.2s and for $Cl_t^{\geqslant} = 0.21s$. Iranian churn data has $Cl_t^{\leqslant}$ = 0.31s and for $Cl_t^{\geqslant} = 0.31s$. For waveform database generator version 1, $Cl_t^{\leqslant}$ = 0.29s and for $Cl_t^{\geqslant} = 0.3s$. Letter recognition has $Cl_t^{\leqslant}$ = 2.39s and for $Cl_t^{\geqslant} = 2.4s$. For lung cancer $Cl_t^{\leqslant}$ = 0.02s and for $Cl_t^{\geqslant} = 0.023s$.

Table 15: Execution time comparison to calculate approximation sets for suggested and traditional $Cl_t^{\geq}$

Dataset	**Time Taken(s)**		**Percentage reduction in execution time**
	Conventional Approximation sets for $\boldsymbol{Cl_t^{\geq}}$	**Proposed Approximation sets for $\boldsymbol{Cl_t^{\geq}}$**	
Breast Cancer Coimbra	1.06	0.024	97.6%
Caesarian Section Classification	20.01	0.03	99.8%
Glass	4.1	0.029	99.2%
Haberman	11.02	0.168	98.5%
Iris	2.6	0.02	99.2%
Concrete Compressive strength	68.57	0.2	99.7%
Iranian Churn Data	102.5	0.31	99.6%
Waveform database generator version 1	121.1	0.29	99.7%
Letter Recognition	330.15	2.39	99.2%
Lung Cancer	0.17	0.02	88.2%

In table 15, we have compared execution time to compute approximation sets for conventional and proposed $Cl_t^{\geq}$. For that we have used 10 UCI datasets. We applied conventional approach and proposed approach to computer approximation for more preferred class. And we have got remarkable results that showed us that proposed approach take lesser time as compare to conventional approach. The graphical representation of percentage decrease in execution time to compute $Cl_t^{\geq}$ shown below in figure 11.

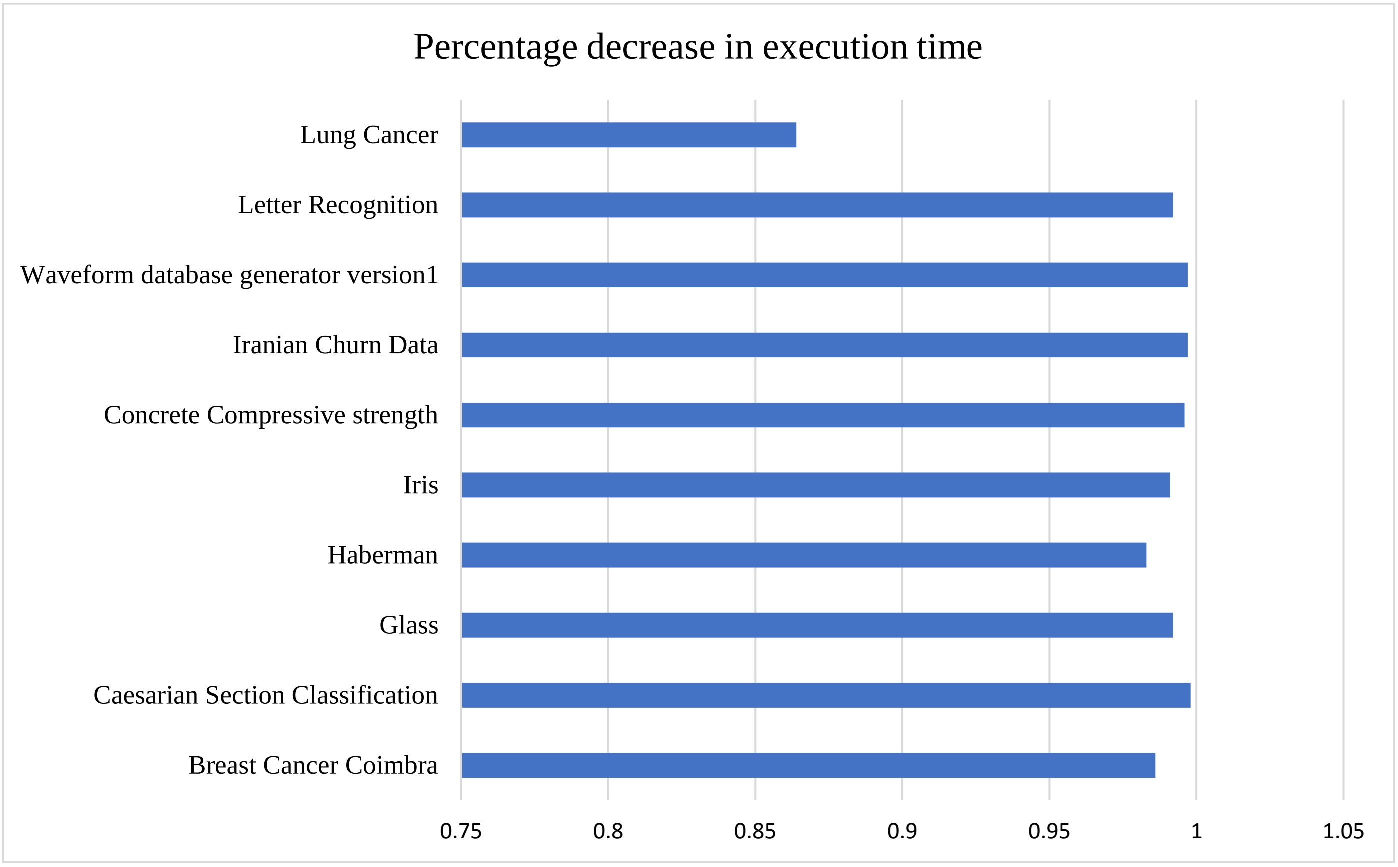


Figure 12: Percentage decrease in execution time for $Cl_t^{\geq}$

Table 16: Execution time comparison to calculate approximation sets for suggested and traditional $Cl_t^{\leq}$

Dataset	**Time duration (s)**		**Percentage decrease in execution time**
	Conventional sets of Approximation for $Cl_t^{\leq}$ (s)	**Proposed sets of Approximation for $Cl_t^{\leq}$ (s)**	
Breast Cancer Coimbra	1.91	0.026	98.6%
Caesarian Section Classification	20.8	0.03	99.8%
Glass	3.9	0.029	99.2%
Haberman	10.33	0.17	98.3%
Iris	2.4	0.021	99.1%
Concrete Compressive strength	70.8	0.22	99.6%
Iranian Churn Data	111.25	0.32	99.7%
Waveform database generator version1	124.44	0.29	99.7%
Letter Recognition	324.86	2.66	99.2%
Lung Cancer	0.17	0.023	86.4%

In table 16, we have shown comparison of execution time for both conventional and proposed algorithm to compute $Cl_t^{\geq}$. We have used 10 UCI datasets. We applied conventional approach and proposed approach to computer approximation for less preferred class. Results shows remarkable decrease in computational time. We have seen that execution time of proposed approach is far lesser than that of conventional approach. Average percentage reduction in execution time is shown in figure 13 below:

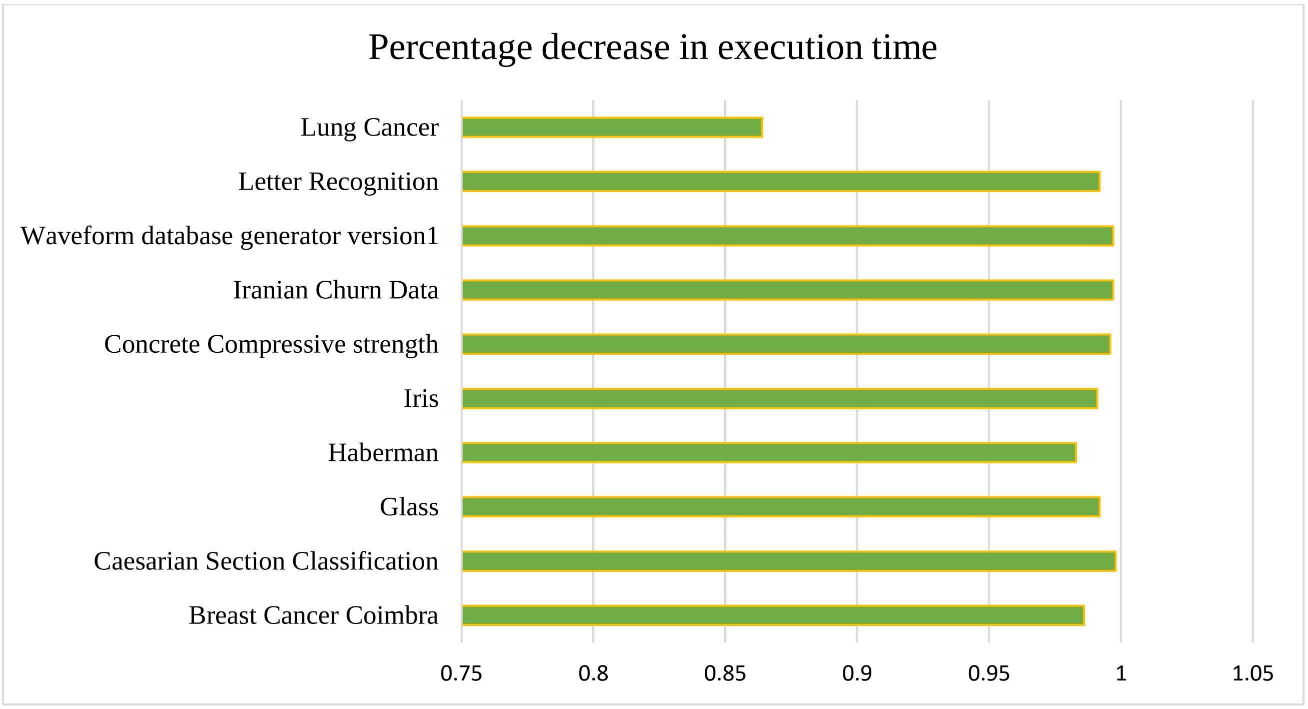


Figure 13: Percentage decrease in execution time for $Cl_t^{\leq}$

Table 17: Time comparison using recent techniques

Dataset	**Time duration taken(s)**		
	An improved technique for estimating DRSA approximations (Ahmad et al, 2022)	**A revised method for estimating DRSA approximations (Nosheen et al, 2022b)**	**Proposed Methodology**
Gisette	28.7	20.01	11.82
EEG Eye State	12.1	9.5	6.1
URL Reputation	77.6	51.2	32.6
P53 Mutants	19.3	15.1	7.75
Average decrease in comparison to the Traditional DRSA	70%	83%	99.2%

Table 17 compares the time required by the suggested method with more modern approaches. We have used the "redefined method for computing approximations of DRSA" and the "improved technique to compute approximations of DRSA," two current methodologies. Four datasets have been taken into consideration: P53 mutants, URL reputation, EEG eye state, and Gisette. Table 17 provides the average decrease when compared to the traditional DRSA, and Figure 13 displays a graphical depiction of that reduction. The suggested approach has the longest decrease time when compared to alternative methods, according to the results.

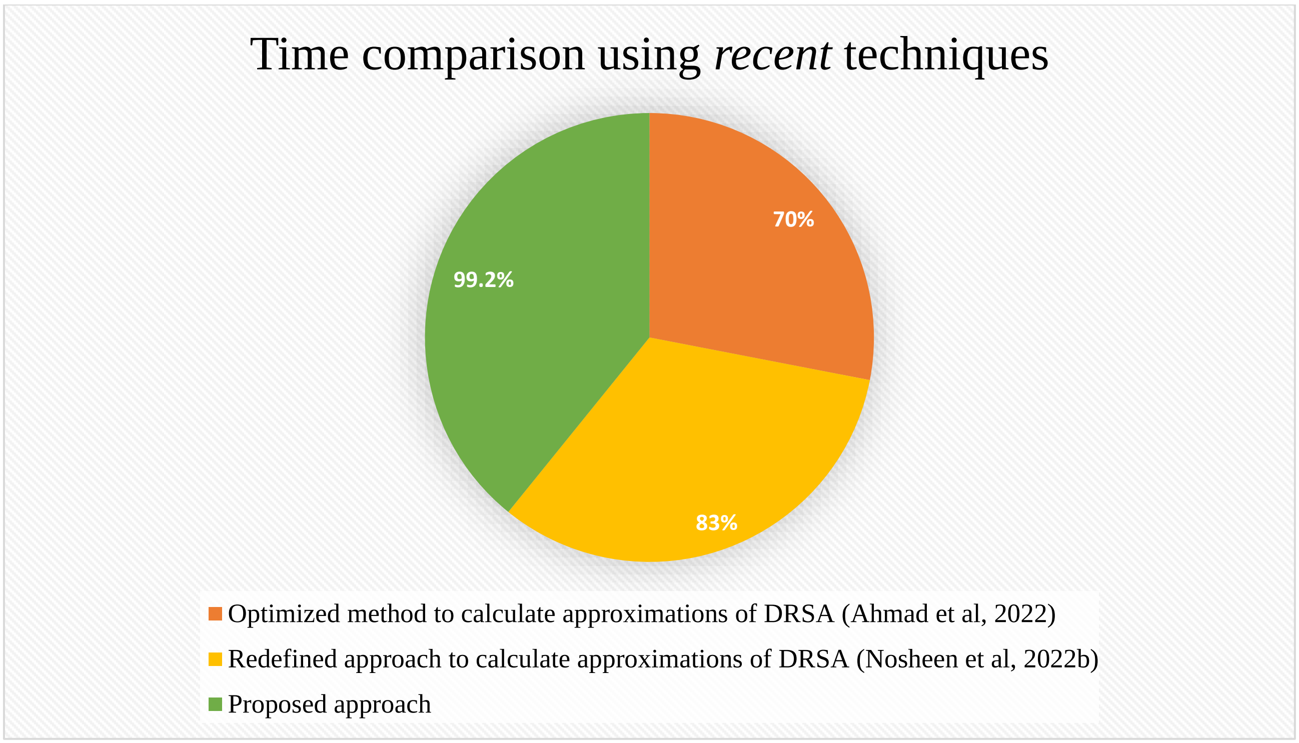


Figure 14: Time comparison using recent techniques

6.5.1 Computational Time

Using the aforementioned hardware and software requirements, we have executed the proposed, current parallel, and traditional algorithms on 10 UCI datasets and compared their execution times. (See table 12 for a mention). We measured the amount of time needed to compute approximations for the upper union of classes and the lower union of classes separately using the standard approach. (Mentioned in Table 13).

In the suggested technique to compute approximations with minimum computation, we developed KD tree rules to compute lower and higher approximations for the upward and downward union of classes. Thus, by using the suggested method, we saw a notable decrease in the execution time, as seen in table 14. To evaluate the effectiveness of both strategies, we computed the percentage savings in time taken to execute using the procedure provided in Eq. (10). he two techniques are compared in Tables 15–16. For downward union of classes $Cl_t^{\leq}$ and upward union of classes $Cl_t^{\geq}$ percentage reduction in execution time is almost 98%.

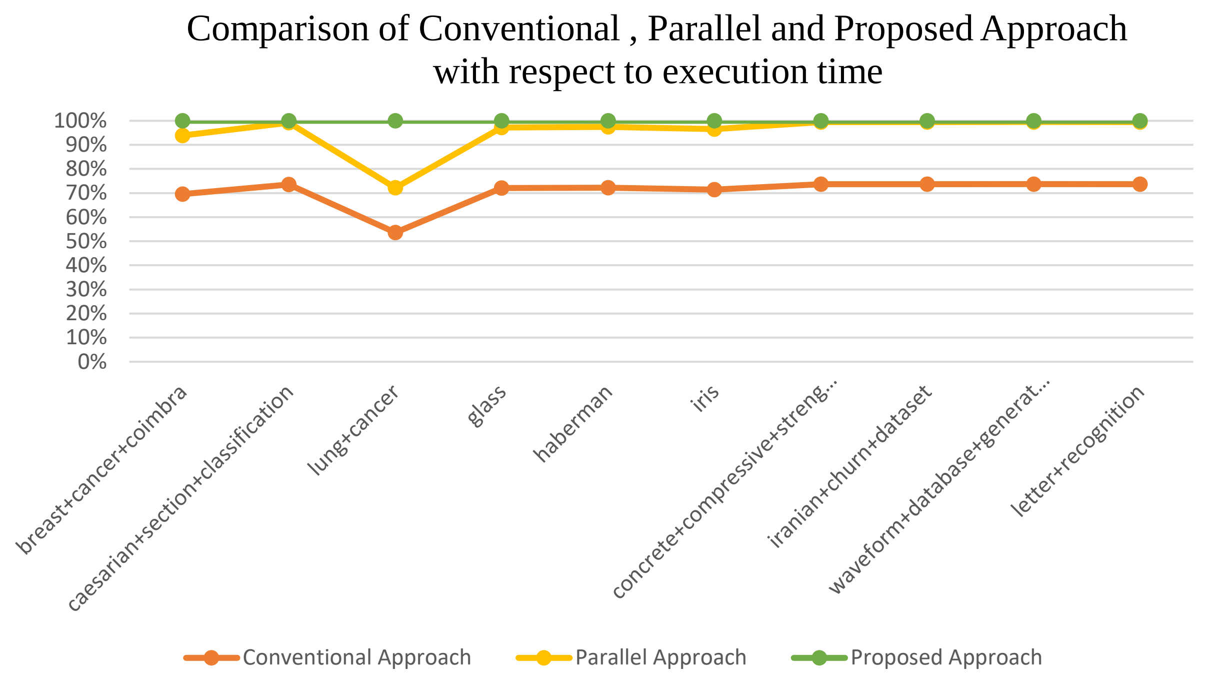


Figure 15: Comparison of Conventional, Parallel and Proposed Approach with respect to execution time

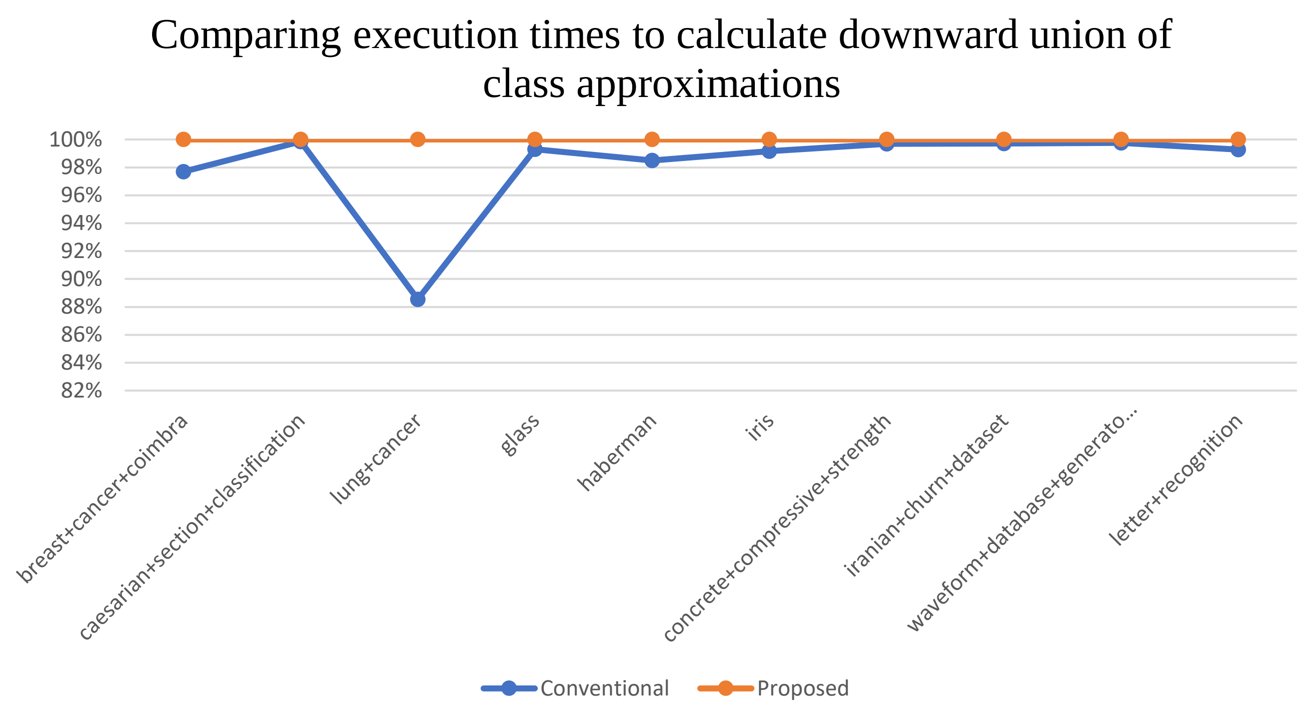


Figure 16: Comparing execution times to calculate downward union of class approximations

In our research, we have conducted a thorough analysis of the proposed approach and the conventional approach of dominance-based rough set theory on ten UCI datasets. Through meticulous calculations and comparisons, it has become evident that our proposed algorithm offers a significant advantage in terms of reduced execution time when compared to the conventional approach.

Furthermore, to visually demonstrate the superiority of my approach, we have created a graphical representation that clearly illustrates the performance difference between the two methods. The graph showcases a noticeable decrease in execution time when utilizing my proposed algorithm, further solidifying its superiority.

These results underscore the importance of adopting my approach in the field of dominance-based rough set theory. The reduced execution time not only enhances computational efficiency but also opens up possibilities for real-time applications and large-scale data processing. By leveraging the power of our algorithm, researchers and practitioners can achieve more efficient and effective outcomes in their respective domains.

Table 18: Comparison of memory consumption

Dataset	Conventional Approach (bytes)	Proposed Approach (bytes)
Breast Cancer Coimbra	9600000	8900000
Caesarian Section Classification	15250000	9780000
Glass	13680000	9250000
Haberman	70800000	10980000
Iris	20880000	9030000
Concrete Compressive strength	210000000	11190000
Iranian Churn Data	300000000	11950000
Waveform database generator version 1	303900000	12010000
Letter Recognition	691500000	18150000
Lung Cancer	8900000	8820000

Table 18 that shows the memory consumption of conventional DRSA technique and proposed approach. Memory consumption was calculated in bytes which is a memory unit. Proposed approach took less memory as compare to conventional approach. We have used 10 UCI datasets for this. The average reduction percentage for memory consumption is 93.3% using equation 10.

6.5.2 Memory Consumption

The dominance-based rough set technique has drawn a lot of interest in the field of data analysis and decision-making because of its ability to handle imprecise and uncertain data. The effective management of memory is one of the main obstacles to putting this strategy into practice, especially when working with enormous datasets. But compared to the traditional method, a considerable memory reduction has been made by adding the KD-tree data structure.

The binary search tree known as the KD-tree, divides data points in a multidimensional space. It creates a hierarchical data structure that makes it possible to conduct effective search and retrieval operations. The KD-tree optimizes the storing and retrieval of data points when used with the dominance-based rough set technique, resulting in less memory use.

The memory saving advantages of the dominance-based rough set technique are revealed by using the KD-tree. The KD-tree's hierarchical structure makes search operations quicker and more effective, allowing the method to handle bigger datasets without suffering performance penalties. The technique is made more effective overall thanks to the decrease in memory utilization, which also makes it possible to analyses larger and more complicated information, producing more accurate and trustworthy findings.

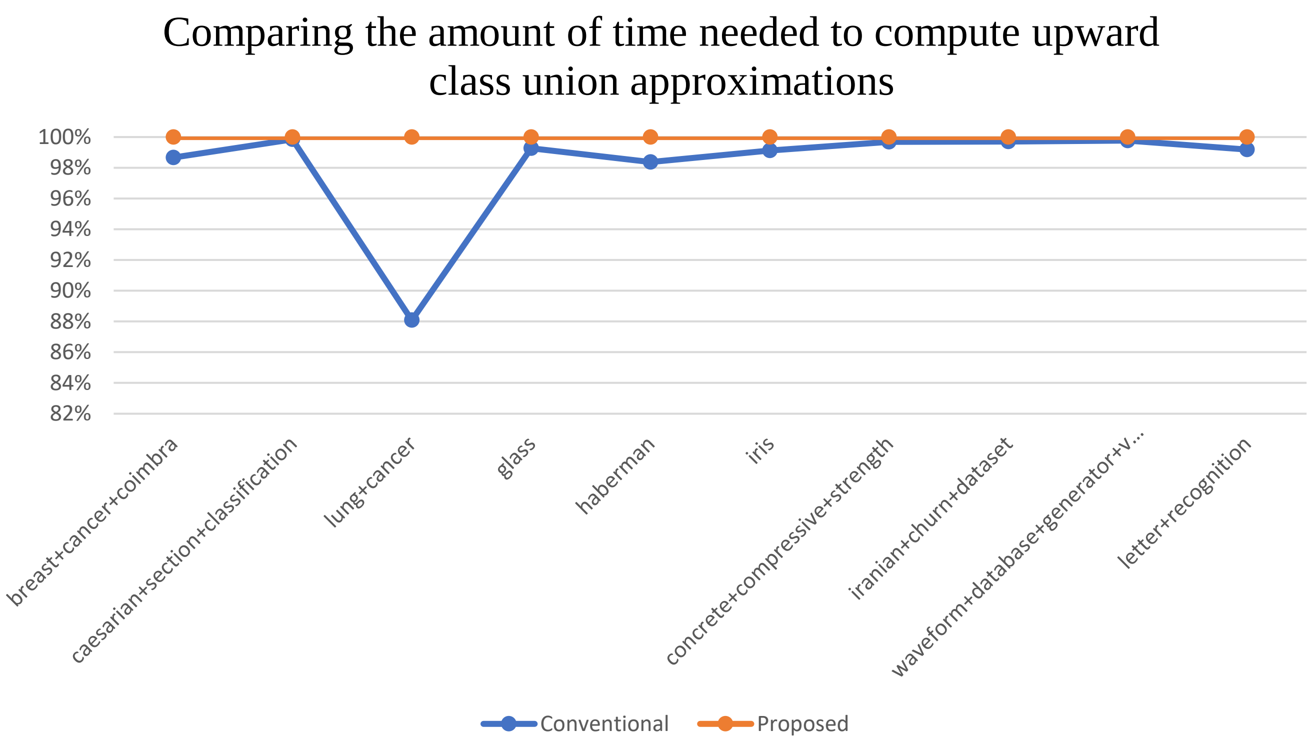


Figure 17: Comparing the amount of time needed to compute upward class union approximations

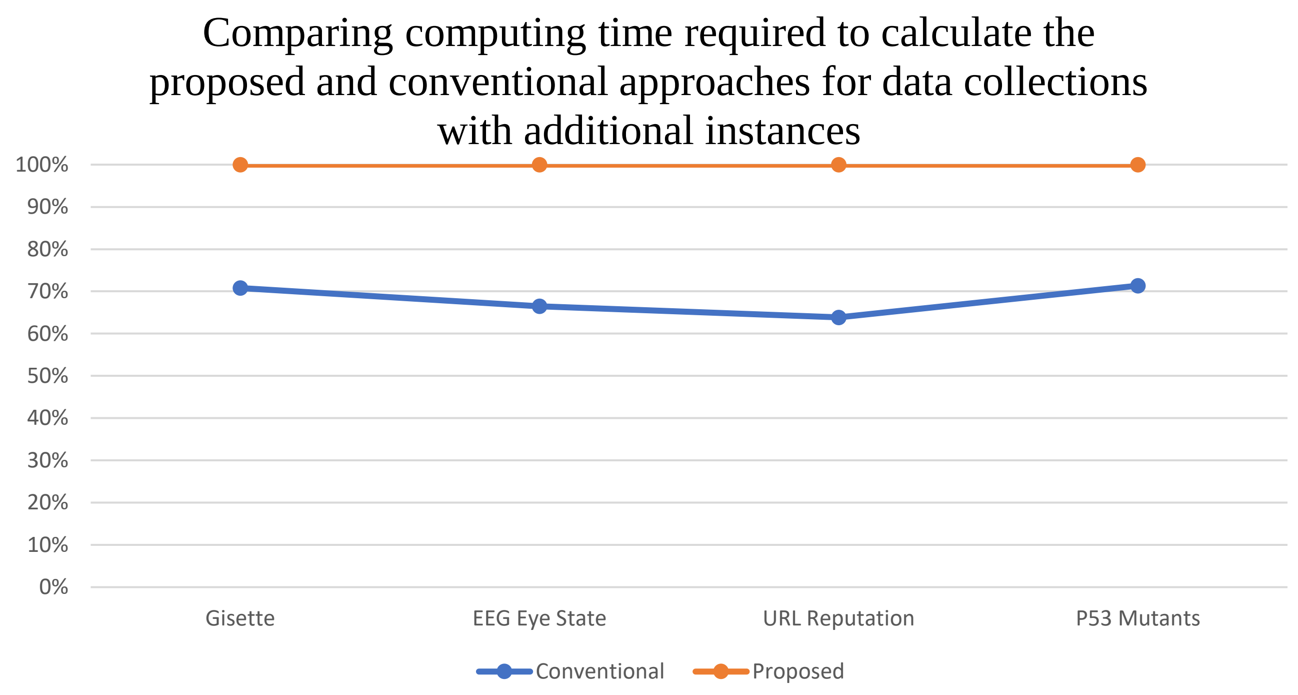


Figure 18: Comparing computing Time required to calculate the proposed and conventional approaches for data collections with additional instances

In our research, we have conducted a thorough analysis of the proposed approach and the conventional approach of dominance-based rough set theory on ten UCI datasets. Through meticulous calculations and comparisons, it has become evident that my proposed algorithm offers a significant advantage in terms of reduced execution time when compared to the conventional approach. This finding highlights the efficiency and time-saving potential of my approach.

Furthermore, to visually demonstrate the superiority of my approach, we have created a graphical representation that clearly illustrates the performance difference between the two methods. The graph showcases a noticeable decrease in execution time when utilizing my proposed algorithm, further solidifying its superiority. As shown in figure 16 and 17.

These results underscore the importance of adopting my approach in the field of dominance-based rough set theory. The reduced execution time not only enhances computational efficiency but also opens up possibilities for real-time applications and large-scale data processing. By leveraging the power of my algorithm, researchers and practitioners can achieve more efficient and effective outcomes in their respective domains.

6.6 Case Study

The shift in the modern period towards technology and the requirement for efficient data processing of large datasets provide considerable obstacles. In order to overcome these difficulties, the Dominance-based Rough Set Approach (DRSA), an improved method within the rough set theory, finds significant data in preference-ordered datasets. However, it can be expensive to compute lower and higher approximations in DRSA, particularly if the data varies over time. In order to solve this problem, the proposed approach efficiently computes estimates for increasing object values. Results show that this strategy is efficient and effective, with significant decreases in execution time, memory usage, and structural complexity when compared to traditional approaches using publicly accessible datasets from UCI.

Our suggested methodology performs better than current methods in terms of accuracy and efficiency, in addition to significantly lowering execution time for the DRSA issue. Our technique exhibits excellent performance through thorough comparison analysis, offering a considerable improvement in addressing the issues of DRSA. It is an excellent option for researchers and practitioners looking for the best answers in this subject because of its unique methodology and strong outcomes.

CHAPTER 7 : CONCLUSION AND FUTURE WORK

This chapter will wrap up our efforts and provide some future advice. Section 7.1 presents the findings of our study endeavor, and Section 7.2 outlines our plans for the future. Additionally, section 7.3 mentions limitations.

7.1 Conclusion

The shift in the modern period towards technology and the requirement for efficient data processing of large datasets provide considerable obstacles. In order to overcome these difficulties, the Dominance-based Rough Set Approach (DRSA), an improved method within the rough set theory, finds significant data in preference-ordered datasets. However, it can be expensive to compute lower and higher approximations in DRSA, particularly if the data varies over time. In order to solve this problem, the proposed approach efficiently computes estimates for increasing object values.

Results show that this strategy is efficient and effective, with significant decreases in execution time, memory usage, and structural complexity when compared to traditional approaches using publicly accessible datasets from UCI.

Our suggested methodology performs better than current methods in terms of accuracy and efficiency, in addition to significantly lowering execution time for the DRSA issue. Our technique exhibits excellent performance through thorough comparison analysis, offering a considerable improvement in addressing the issues of DRSA. It is an excellent option for researchers and practitioners looking for the best answers in this subject because of its unique methodology and strong outcomes.

The modern era's reliance on technology, as well as the need for fast processing of large datasets, provide considerable obstacles. To address these challenges, the Dominance-based Rough Set Approach (DRSA), an advanced approach in rough set theory, emerges as a solution for discovering critical data in preference-ordered datasets. However, the computational cost of calculating lower and higher approximations in DRSA, particularly when working with dynamic data, might be prohibitively expensive. To solve this issue, we present an efficient method for predicting growing object values. The results reveal that this technique is successful, with significant savings in execution time, memory use, and structural complexity compared to standard approaches, as proven by testing on publicly accessible UCI datasets.

Our technique not only surpasses previous methods in terms of accuracy and efficiency, but it also drastically decreases the execution time required for the DRSA issue. Through detailed comparison research, our method displays remarkable performance, indicating a significant improvement in tackling DRSA issues. With its novel methodology and appealing results, our approach is a potential alternative for academics and practitioners looking for the best solutions in this field.

7.2 Future work

Theory stands out as a far more efficient and time-saving solution compared to the conventional approach. The evidence from the calculations, as well as the graphical representation, supports the claim that my approach offers significant advantages in terms of execution time reduction. Its potential impact on various applications and domains cannot be overlooked, making it a promising avenue for future research and implementation.

7.3 Limitation

The limitation of our proposed work is we have used KD tree approach which has the disadvantage of becoming less efficient as the complexity of the data rises. This is referred to as the "curse of dimensionality." The splitting procedure gets less efficient as the number of dimensions increases, producing a less balanced tree and perhaps affecting the algorithm's performance.

References

[1] Bhattarai, B. P., Paudyal, S., Luo, Y., Mohanpurkar, M., Cheung, K., Tonkoski, R., ... & Zhang, X. (2019). Big data analytics in smart grids: state-of-the-art, challenges, opportunities, and future directions. *IET Smart Grid*, *2*(2), 141-154.

[2] Himanen, L., Geurts, A., Foster, A. S., & Rinke, P. (2019). Data-driven materials science: status, challenges, and perspectives. *Advanced Science*, *6*(21), 1900808.

[3] Wang, X., Wu, J., Chen, J., Li, L., Wang, Y. F., & Wang, W. Y. (2019). Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 4581-4591).

[4] Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and information technologies*, *19*(1), 3-26.

[5] Brownlee, J. (2020). *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery.

[6] Bolón-Canedo, V., & Alonso-Betanzos, A. (2019). Ensembles for feature selection: A review and future trends. *Information Fusion*, *52*, 1-12.

[7] Di Mauro, M., Galatro, G., Fortino, G., & Liotta, A. (2021). Supervised feature selection techniques in network intrusion detection: A critical review. *Engineering Applications of Artificial Intelligence*, *101*, 104216.

[8] Jiang, T., Gradus, J. L., & Rosellini, A. J. (2020). Supervised machine learning: a brief primer. *Behavior Therapy*, *51*(5), 675-687.

[9] Roffo, G., Melzi, S., Castellani, U., Vinciarelli, A., & Cristani, M. (2020). Infinite feature selection: a graph-based feature filtering approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(12), 4396-4410.

[10] Bommert, A., Sun, X., Bischl, B., Rahnenführer, J., & Lang, M. (2020). Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, *143*, 106839.

[11] Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and information technologies*, *19*(1), 3-26.

[12] Alzaqebah, M., Alrefai, N., Ahmed, E. A., Jawarneh, S., & Alsmadi, M. K. (2020). Neighborhood search methods with moth optimization algorithm as a wrapper method for

feature selection problems. *International Journal of Electrical & Computer Engineering (2088-8708)*, *10*(4).

[13] González, J., Ortega, J., Damas, M., Martín-Smith, P., & Gan, J. Q. (2019). A new multi-objective wrapper method for feature selection–Accuracy and stability analysis for BCI. *Neurocomputing*, *333*, 407-418.

[14] Zhang, J., Xiong, Y., & Min, S. (2019). A new hybrid filter/wrapper algorithm for feature selection in classification. *Analytica chimica acta*, *1080*, 43-54.

[15] Liu, H., Zhou, M., & Liu, Q. (2019). An embedded feature selection method for imbalanced data classification. *IEEE/CAA Journal of Automatica Sinica*, *6*(3), 703-715.

[16] Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and information technologies*, *19*(1), 3-26.

[17] Solorio-Fernández, S., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2020). A review of unsupervised feature selection methods. *Artificial Intelligence Review*, *53*(2), 907-948.

[18] Brownlee, J. (2019). How to choose a feature selection method for machine learning. *Machine Learning Mastery*, *10*.

[19] Tang, M., & Liao, H. (2021). From conventional group decision making to large-scale group decision making: What are the challenges and how to meet them in big data era? A state-of-the-art survey. *Omega*, *100*, 102141.

[20] Meng, T., Jing, X., Yan, Z., & Pedrycz, W. (2020). A survey on machine learning for data fusion. *Information Fusion*, *57*, 115-129.

[21] Pawlak, Zdzisław. "Rough sets." International journal of computer & information sciences 11 (1982): 341-356.

[22] Pawlak, Zdzislaw, et al. "Rough sets." *Communications of the ACM* 38.11 (1995): 88-95.

[23] Pawlak, Zdzisław, and Andrzej Skowron. "Rudiments of rough sets." *Information sciences* 177.1 (2007): 3-27.

[24] Greco, S., Matarazzo, B., Słowiński, R.: Rough sets theory for multi-criteria decision analysis. European Journal of Operational Research, **129**, 1 (2001) 1–47

[25] Greco, S., Matarazzo, B., Słowiński, R.: Multicriteria classification by dominance-based rough set approach. In: W.Kloesgen and J.Zytkow (eds.), Handbook of Data Mining and Knowledge Discovery, Oxford University Press, New York, 2002

[26] Słowiński, R., Greco, S., Matarazzo, B.: Rough set based decision support. Chapter 16 [in]: E.K. Burke and G. Kendall (eds.), Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, Springer-Verlag , New York (2005) 475–527

[27] Liou, James JH, and Gwo-Hshiung Tzeng. "A dominance-based rough set approach to customer behavior in the airline market." *Information Sciences* 180.11 (2010): 2230-2238.

[28] Yang, Xibei, et al. "Dominance-based rough set approach and knowledge reductions in incomplete ordered information system." *Information Sciences* 178.4 (2008): 1219-1234.

[29] Chakhar, Salem, et al. "Dominance-based rough set approach for group decisions." *European Journal of Operational Research* 251.1 (2016): 206-224.

[30] Greco, Salvatore, Benedetto Matarazzo, and Roman Słowiński. "Dominance-based rough set approach as a proper way of handling graduality in rough set theory." *Transactions on Rough Sets VII: Commemorating the Life and Work of Zdzisław Pawlak, Part II* (2007): 36-52

[31] Li, Shaoyong, Tianrui Li, and Dun Liu. "Dynamic Maintenance of Approximations in Dominance-Based Rough Set Approach under the Variation of the Object Set." *International Journal of Intelligent Systems* 28, no. 8 (2013): 729-751.

[32] Chen, Hongmei, et al. "A rough-set-based incremental approach for updating approximations under dynamic maintenance environments." *IEEE Transactions on Knowledge and Data Engineering* 25.2 (2011): 274-284.

[33] Cheng, Yi. "The incremental method for fast computing the rough fuzzy approximations." *Data & Knowledge Engineering* 70.1 (2011): 84-100.

[34] Li, Shaoyong, Tianrui Li, and Dun Liu. "Dynamic Maintenance of Approximations in Dominance-Based Rough Set Approach under the Variation of the Object Set." *International Journal of Intelligent Systems* 28.8 (2013): 729-751.

[35] Luo, Chuan, Tianrui Li, and Hongmei Chen. "Dynamic maintenance of approximations in set-valued ordered decision systems under the attribute generalization." *Information Sciences* 257 (2014): 210-228.

[36] Luo, Chuan, et al. "Fast algorithms for computing rough approximations in set-valued decision systems while updating criteria values." *Information Sciences* 299 (2015): 221-242.

[37] Wang, Shu, et al. "Efficient updating rough approximations with multi-dimensional variation of ordered data." *Information Sciences* 372 (2016): 690-708.

[38] Chen, Hongmei, Tianrui Li, and Da Ruan. "Maintenance of approximations in incomplete ordered decision systems while attribute values coarsening or refining." *Knowledge-Based Systems* 31 (2012): 140-161.

[39] Liu, D., Li, T., Ruan, D. and Zou, W., 2009. An incremental approach for inducing knowledge from dynamic information systems. *Fundamenta Informaticae*, *94*(2), pp.245-260.

[40] Li, Shaoyong, et al. "Parallel computing of approximations in dominance-based rough sets approach." *Knowledge-Based Systems* 87 (2015): 102-111.

[41] Raza, Muhammad Summair, and Usman Qamar. "A parallel approach to calculate lower and upper approximations in dominance based rough set theory." *Applied Soft Computing* 84 (2019): 105699.

[42] Nosheen, Faryal, Usman Qamar, and Muhammad Summair Raza. "A parallel rule-based approach to compute rough approximations of dominance based rough set theory." *Engineering Applications of Artificial Intelligence* 115 (2022): 105285.

[43] Li, Shaoyong, and Tianrui Li. "A parallel matrix-based approach for computing approximations in dominance-based rough sets approach." *Rough Sets and Knowledge Technology: 9th International Conference, RSKT 2014, Shanghai, China, October 24-26, 2014, Proceedings 9*. Springer International Publishing, 2014.

[44] Zhang, Junbo, et al. "A parallel method for computing rough set approximations." *Information Sciences* 194 (2012): 209-223.

[45] Li, Shaoyong, et al. "Parallel computing of approximations in dominance-based rough sets approach." *Knowledge-Based Systems* 87 (2015): 102-111.

[46] Qian, Jin, Ping Lv, Xiaodong Yue, Caihui Liu, and Zhengjun Jing. "Hierarchical attribute reduction algorithms for big data using MapReduce." Knowledge-Based Systems 73 (2015): 18-31.

[47] UCI Machine Learning Repository [http://archive.ics.uci.edu/ml].

[48] **Citation of prisma 2009 flowchart:** Rethlefsen, M. L., & Page, M. J. (2022). PRISMA 2020 and PRISMA-S: common questions on tracking records and the flow diagram. *Journal of the Medical Library Association: JMLA*, *110*(2), 253.

[49] Guo, Z., Liu, H., Shi, H., Li, F., Guo, X., & Cheng, B. (2023). KD-Tree-Based Euclidean Clustering for Tomographic SAR Point Cloud Extraction and Segmentation. *IEEE Geoscience and Remote Sensing Letters*, *20*, 1-5.

[50] Şenol, A. (2023). MCMSTClustering: defining non-spherical clusters by using minimum spanning tree over KD-tree-based micro-clusters. *Neural Computing and Applications*, *35*(18), 13239-13259.

[51] Tiwari, V. R. Developments in KD Tree and KNN Searches. *International Journal of Computer Applications*, *975*, 8887.

[52] Ponnusamy, P. P., Shabariram, C. P., Umayal, V. R., & Susmeta, A. (2023, January). Closest Celestial Body Search Using KD Trees. In *2023 International Conference on Computer Communication and Informatics (ICCCI)* (pp. 1-7). IEEE.

[53] Zhang, H., Xu, Y., Liu, Q., Wang, X., & Li, Y. (2022). Solving Fokker–Planck equations using deep KD-tree with a small amount of data. *Nonlinear Dynamics*, *108*(4), 4029-4043.

[54] Shan, Y., Li, S., Li, F., Cui, Y., Li, S., Zhou, M., & Li, X. (2022). A density peaks clustering algorithm with sparse search and Kd tree. *IEEE Access*, *10*, 74883-74901.

[55] Dinh, N. T., Le, T. M., & Van, T. T. (2022, April). An Improvement Method of Kd-Tree Using k-Means and k-NN for Semantic-Based Image Retrieval System. In *World Conference on Information Systems and Technologies* (pp. 177-187). Cham: Springer International Publishing.

[56] Mousa, M. H., & Hussein, M. K. (2022). Toward high-performance computation of surface approximation using a GPU. *Computers and Electrical Engineering*, *99*, 107761.

[57] Zheng, Z., Zha, B., Zhou, Y., Huang, J., Xuchen, Y., & Zhang, H. (2022). Single-stage adaptive multi-scale point cloud noise filtering algorithm based on feature information. *Remote sensing*, *14*(2), 367.

[58] Mir, M., Yaghoobi, M., & Khairabadi, M. (2023). A new approach to energy-aware routing in the Internet of Things using improved Grasshopper Metaheuristic Algorithm with Chaos theory and Fuzzy Logic. *Multimedia Tools and Applications*, *82*(4), 5133-5159.