iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** IV **Month of publication:** April 2020

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

Dynamic Item Set Counting Implication Rules for Student Application Data

N. Indira[1], P. Niruba[2], S. Sruthi[3], V. Swetha[4]
[1]*Faculty Member,* [2, 3, 4]*UG Student, Dept. of CSE, Panimalar Engineering College, Tamil Nadu, India*

Abstract: ***Data cleaning in data mining is the process of detecting and removing corrupt or inaccurate records from a record set, table or database. It improves the data quality and in doing so, increases overall productivity. On cleaning the data, all outdated or incorrect information is gone leaving the highest quality information. After cleaning, a dataset should be uniform with other related datasets in the operation. Here the Admin takes in charge of the student database where the database may include dirty data such as missing data or incorrect data. Due to these unwanted data, there can be an excessive number of data in the dataset and this might even fill up the space. This Data Removal process takes place in two stages. In first stage, all the null values or missing data will be removed and will be stored in Analyzed Data. In second stage, the duplicate data or the repeated data will be eliminated and stored in Original Data. We display that this technique identifies the defective data with high exactness. Examination on clear information displays that our fix strategy secures awesome fixes without presenting them. This original data which is the final filtered data will be displayed to the User. Therefore the cleaned dataset can be viewed after the eradication of duplicate records of data.***

Keywords*: *Data cleansing, Analyzed data, Original data

I. INTRODUCTION

Data cleaning, also called data scrubbing, data cleansing, or data preparation, is the act of taking collected data and making it usable in the preferred statistical software. Cleaning includes removing bad data, creating correct labels and codes, and for everything to be consistent. It's sometimes unavoidable to collect data that are not consistent. It's uncommon for data to be collected and to instantly be ready in a useable state. Almost every analyst and researcher – whether in the field or academia – has data that needs cleaning, so there's no need to ever feel like you did a bad job with the collection stage.

In late years, recognizing inconsistencies in data has focused on a constraint based data quality philosophy: a ton of necessities in some cognizant formalism is connected with a database, and the data is seen as unsurprising or clean if and just if all objectives are satisfied. Various such formalisms exist, getting a wide arrangement of abnormalities, and conscious strategies for fixing the distinguished anomalies are set up. The goals are either given by experts, or thus found from the data. When set up, they are treated as a best quality level, both complete and right, and are thusly used for fixing the data with the true objective that the fix satisfies the necessities. Accept that the used necessities were found on the muddled data, by then this data is right now changed by fixing. Everything considered, running the prerequisite disclosure figuring again on the fix would achieve the divulgence of different impediments and bungles. Toward the day's end, using the basis from already, the fix isn't so impeccable everything considered. Indeed, it does no longer suffice to only get to the bottom of inconsistencies for constraints found on the authentic soiled data , one also has to ensure that no new constraints (and thus new inconsistencies) are observed on the repaired data. Although established constraint formalisms ought to be considered, it is addressed that these challenges for new error detection formalism, referred to as forbidden item sets. In a nutshell, forbidden item sets seize unlikely value co-occurrences where unlikeness is measured the use of the raise measure, generally used in item set mining.

II. DATA CLEANING PROCESS

In this paper, a fixing count is proposed that is eagerly joined to restriction exposure. In particular, it is used to find fixes that are immaculate with respect to an amazing idea of data quality in which a database is regarded to be immaculate if a basic disclosure estimation does not recognize any harmed prerequisites on that data. In separate, existing work based on a static idea of data quality in which an ideal database is required to satisfy a great deal of given constraints. Essentially, cleaning data methodology does not require re-running the exposure method after the fixing step. The notion of forbidden item sets has been added, representing inconsistencies in the statistics as unlawful price combinations. The problem in full generality is described in terms of

a constraint language and a likeliness function, before specializing this putting for low-lift forbidden item sets. This paper only reflects on consideration on the concrete case of low-lift forbidden item sets. Intuitively, the carry measure offers an indication of how probably the co-occurrence of a set of items is given their separate frequencies. It first furnishes motivation for thinking about invalid or unlikely value combos (formalized as low lift forbidden item sets) as error detection formalism.

Fig 2.1 Data Cleaning Process

III. RELATED WORK

A. For entity resolution, it remains very challenging to find the solution with quality guarantees as measured by both precision and recall. In the year 2017, the authors Zhaoqiang Chen , Qun Chen and Zhanhuai Li proposed a HUman-and-Machine cooperative framework, denoted by HUMO, for entity resolution. Compared with the existing approaches, HUMO enables a flexible mechanism for quality control that can enforce both precision and recall levels. They also introduced the problem of minimizing human cost given a quality requirement and present corresponding optimization techniques. Finally, they demo that HUMO achieves high-quality results with reasonable return on investment (ROI) in terms of human cost on real datasets.

B. Data quality is one of the most important problems in data management, since dirty data often leads to inaccurate data analytics results and wrong business decisions. According to a report by Insight Squared in 2012, poor data across businesses and the government cost the United States economy 3.1 trillion dollars a year. To detect data errors, data quality rules or integrity constraints (ICs) have been proposed by Ihab F. Ilyas and Xu Chu as a declarative way to describe legal or correct data instances in the year 2015. Any subset of data that does not conform to the defined rules is considered erroneous, which is also referred to as a violation. Various kinds of data repairing techniques with different objectives have been introduced where algorithms are used to detect subsets of the data that violate the declared integrity constraints, and even to suggest updates to the database such that the new database instance conforms to these constraints. While some of these algorithms aim to minimally change the database, others involve human experts or knowledge bases to verify the repairs suggested by the automatic repeating algorithms. Trends in Cleaning Relational Data: Consistency and Deduplication discusses the main facets and directions in designing error detection and repairing techniques. It proposes taxonomy of current anomaly detection techniques, including error types, the automation of the detection process, and error propagation. It also sets out taxonomy of current data repairing techniques, including the repair target, the automation of the repair process, and the update model. It concludes by highlighting current trends in "big data" cleaning.

C. In the year 2007,the authors Philip Bohannon ;Wenfei Fan ; Floris Geerts ; Xibei Jia ; Anastasios Kementsietsid is proposed a class of constraints, referred to as conditional functional dependencies (CFDs), and study their applications in data cleaning. In contrast to traditional functional dependencies (FDs) that were developed mainly for schema design, CFDs aim at capturing the consistency of data by incorporating bindings of semantic ally related values. For CFDs they provide an inference system analogous to Armstrong's axioms for FDs, as well as consistency analysis. Since CFDs allow data bindings, a large number of individual constraints may hold on a table, complicating detection of constraint violations. They developed techniques for detecting CFD violations in SQL as well as novel techniques for checking multiple constraints in a single query. They experimentally evaluated the performance of our CFD-based methods for inconsistency detection. This not only yields a constraint theory for CFDs but is also a step toward a practical constraint-based method for improving data quality.

D. A Unified Model for Data and Constraint Repair represents domain specific rules and relationships that hold over any database instance that accurately reflects the domain. In the year 2011the authors Fei Chiang, Ren´ee J. Miller proposes an alternative approach that considers both data and the existing constraints in the repair process. When an inconsistency occurs, it is no longer clear if there is an error in the data (and the data should be repaired), or if the constraints have evolved (and the constraints should be repaired). In addition, constraint discovery techniques are expensive to compute and they do not consider in their search the existing set of constraints.

E. Data cleaning is an important problem and data quality rules are the most promising way to face it with a declarative approach. Previous work has focused on specific formalisms, such as functional dependencies (FDs), conditional functional dependencies (CFDs), and matching dependencies (MDs), and those have always been studied in isolation. Moreover, such techniques are usually applied in a pipeline or interleaved. In this work they tackle the problem in a novel, unified framework. First, they let users specify quality rules using denial constraints with ad-hoc predicates. This language which was proposed by Xu Chu, Ihab F. Ilyas and Paolo Papotti in the year 2013 subsumes existing formalisms and can express rules involving numerical values, with predicates such as “greater than” and “less than”. More importantly, it exploits the interaction of the heterogeneous constraints by encoding them in a conflict hyper graph. Such holistic view of the conflicts is the starting point for a novel definition of repair context which allows us to compute automatically repairs of better quality w.r.t. previous approaches in the literature. Experimental results on real datasets show that the holistic approach outperforms previous algorithms in terms of quality and efficiency of the repair.

F. Efficient Discovery of the most interesting associations appears highly demanding, as assessing whether an item set is self-sufficient requires consideration of all pair wise partitions of the item set into pairs of subsets as well as consideration of all supersets. Self-sufficient item sets is achieved by identifying which of the three constraints is violated and the item set(s) with respect to which it is violated. In 2014, Geoffrey I. Webb, Jilles Vreeken realized that objective function is very expensive to calculate, requiring assessment of every binary partition of an item set, and may be neither monotone nor anti-monotone. This prevents massive inflation of the number of item sets discovered due to the addition to each productive item set x of items that are statistically independent of the items in x and due to joining multiple item sets irrespective of whether they interact with one another.

G. Methods for cleaning dirty data typically rely on additional information about the data, such as user-specified constraints that specify when a database is dirty. These constraints often involve domain restrictions and illegal value combinations. Traditionally, a database is considered clean if all constraints are satisfied. However, many real-world scenarios’ only have a dirty database available. In the year 2017, Joeri Rammelaere , Floris Geerts and Bart Goethals adopted a dynamic notion of data quality, in which the data is clean if an error discovery algorithm does not find any errors. They have introduced forbidden itemsets which capture unlikely value co-occurrences in dirty data, and derive properties of the lift measure to provide an efficient algorithm for mining low lift forbidden item sets. They further introduced a repair method which guarantees that the repaired database does not contain any low lift forbidden item sets. The algorithm uses nearest neighbor imputation to suggest possible repairs. Optional user interaction can easily be integrated into the proposed cleaning method. Evaluation on real-world data shows that errors are typically discovered with high precision, while the suggested repairs are of good quality and do not introduce new forbidden item sets, as desired.

IV. DATA ANALYSIS

Data scientists spend a large amount of their time cleaning datasets and getting them down to a form with which they can work. In fact, a lot of data scientists argue that the initial steps of obtaining and cleaning data constitute 80% of the job. Therefore, if you are just stepping into this field or planning to step into this field, it is important to be able to deal with messy data, whether that means missing values, inconsistent formatting, malformed records, or nonsensical outliers.

The absolutely first thing you need to do is to import libraries for data preprocessing. There are lots of libraries available, but the most popular and important Python libraries for working on data are Numpy, Matplotlib, and Pandas. Numpy is the library used for all mathematical things. Pandas is the best tool available for importing and managing datasets. Matplotlib (Matplotlib.pyplot) is the library to make charts. Once the data set is downloaded, name it as a .csv file, then need to load it into a Pandas Data Frame to explore it and perform some basic cleaning tasks removing information you don’t need that will make data processing slower.

DataFrameName.dropna(axis=0,how='any',thresh=None, subset=None, inplace=False)

Pandas DataFrame dropna() function is used to remove rows and columns with Null/NaN values. By default, this function returns a new Data Frame and the source Data Frame remains unchanged.

V. PROPOSED SYSTEM ARCHITECTURE

At whatever point confinement disclosure counts are re-continued running on the fixed data, new impediments and along these lines bungles are as often as possible found. The fixing procedure by then shows new constraint encroachment. A substitute kind of fixing procedure is presented, which abstains from displaying new necessity encroachment, as demonstrated by a revelation count. Forbidden item set is the techniques were the duplicate data will be removed and is the mostly used technique to remove those data. There will be duplicates data from the large dataset.In this approach the duplicates are removed using constraint based cleaning, where the null values are removed by analyzing the data and then the dirty data is removed by constraint based cleaning. Thus the original data is viewed by the user. Unwanted data will be removed and the time taken to remove those duplicate data is very low.

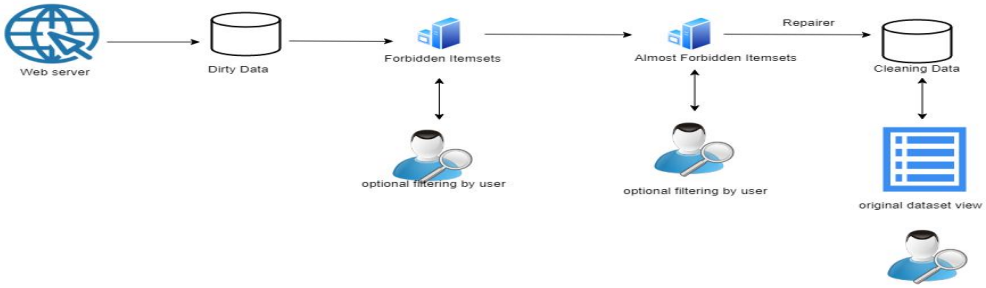


Fig 5.1 System Architecture

A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADL).This diagram gives the overview of the proposed cleaning mechanism. The user/admin will be logged in to the database. The database will contain both required data and unnecessary data. These unwanted data contains the null value, repeated and then the wrong data. For example: If a student comes for an admission to a college, first day if he makes an admission for CSE department and the next day the same student comes and makes an admission for another course. The dirty data in the database will be viewed by the user and it will be optionally removed by using forbidden item sets. Firstly, the null values will be removed and will be present in analyzed data and then the repeated data/unwanted data will be removed. Then the cleaned data will be displayed to the user.

VI. MODULE DESIGN SPECIFICATION

A module is a software component or part of a program that contains one or more routines. One or more independently developed modules make up a program. An enterprise-level software application may contain several different modules, and each module serves unique and separate business operations. Modules make a programmer's job easy by allowing the programmer to focus on only one area of the functionality of the software application. Modules are typically incorporated into the program (software) through interfaces.

A. User Interface Design

This is the first module of the project. The important role for the user is to move login window to user window. This module has created for the security purpose. In this login page the user has to enter login user id and password. It will check username and password is match or not (valid user id and valid password). If they enter any invalid username or password they can't enter into login window to user window it will shows error message. So it prevents unauthorized user from entering into the login window to user window. It will provide a good security for our project. So server contain user id and password server also check the authentication of the user. It well improves the security and preventing from unauthorized user enters into the network. In our project we are using JSP for creating design. Here we validate the login user and server authentication.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

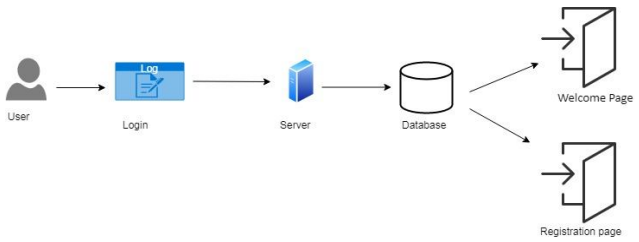


Fig 6.1 User Interface Design

B. Forbidden Item set

In this module, data stored under several databases, from those databases we have to remove unwanted data i.e. duplicate from the storage. Methods for cleaning dirty data typically rely on additional information about the data, such as user-specified constraints that specify when a database is Cleaning Data with Forbidden Item sets**.**

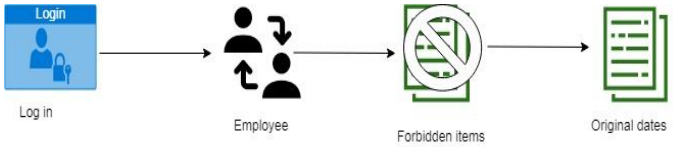


Fig 6.2 Forbidden Item set

C. Clean Dirty Data

In this part the user will remove unwanted data from the databases. Using DISCOVERY algorithm, user will clean unwanted data from the sets. The collection tools and methods can collect the wrong data, or no data. Or worse, they could have issues with unit conversion. It can be seen as (ms) and assume milliseconds, but it could be microseconds.

Fig 6.3 Clean Dirty Data

D. Constraint based Data Cleaning

In this module, original data will retrieved from the above module using discovery algorithm, the unwanted data will be removed or deleted from the set. User will get original content only. While constraints should logically become an essential part of data-cleaning tools, users are not aware of any commercial tools with this functionality. To show that constraints can be effectively used in data cleaning, Semandaq, a research prototype system has been presented for data repairing.

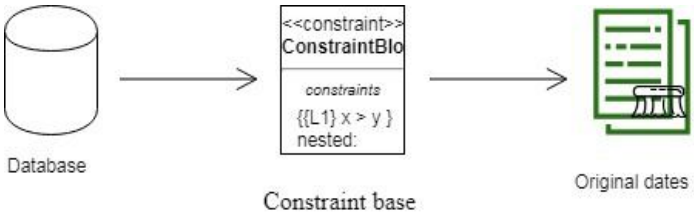


Fig 6.5 Constraint based Data Cleaning

E. View Original Data

Here, user will show the original data, deleted data will not show in databases. Similarly user can get original data without duplicate details or unwanted data.

Fig 6.5 View Original Data

VII. REQUIREMENT ANALYSIS AND SPECIFICATION

These are the requirements for doing the project. Without using these tools and software's the project can't be done. So we have two requirements to do the project. They are

A. Hardware Requirements

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented. The processor used here is Pentium IV 2.6 GHz, Intel Core 2 Duo

1) RAM : 4GB DD RAM
2) MONITOR : 15" COLOR
3) HARD DISK : 40 GB

B. Software Requirements

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

1) FRONT END : J2EE JAVASCRIPT
2) BACK END : MY SQL 5.5
3) OPERATING SYSTEM : Windows 07
4) IDE : Eclipse

VIII. SOFTWARE TESTING

Software testing is about checking if the software works properly and if it meets the written requirements specifications. Types of Software testing are:

A. Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

1) Types of Unit Testing: There are 2 type of Unit Testing:
a) Manual
b) Automated
2) Workflow of Unit Testing

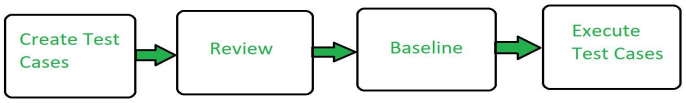


Fig 7.1.1

B. Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

1) Valid Input : identified classes of valid input must be accepted.
2) Invalid Input : identified classes of invalid input must be rejected.
3) Functions : identified functions must be exercised.
4) Output : identified classes of application outputs must be exercised.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue IV Apr 2020- Available at www.ijraset.com

Functional testing is a kind of black-box testing that is performed to confirm that the functionality of an application or system is behaving as expected.
It is done to verify all the functionality of an application

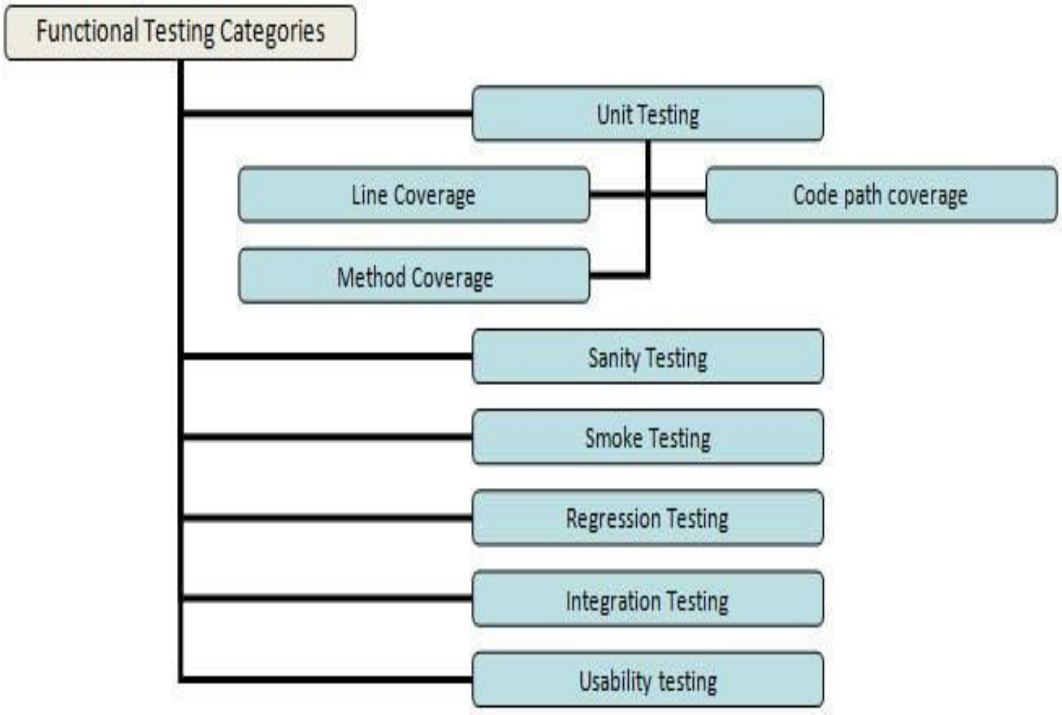


Fig 7.1.2

C. System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. System testing takes, as its input, all of the integrated components that have passed integration testing. The purpose of integration testing is to detect any inconsistencies between the units that are integrated together (called assemblages). System testing seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. The actual result is the behavior produced or observed when a component or system is tested.

D. Performance Testing

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results. The methodology adopted for performance testing can vary widely but the objective for performance tests remain the same. It can help demonstrate that your software system meets certain pre-defined performance criteria. Or it can help compare the performance of two software systems. It can also help identify parts of your software system which degrade its performance.

Fig 7.1.3

E. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error. Integration testing is the process of testing the interface between two software units or module. It's focus on determining the correctness of the interface. The purpose of the integration testing is to expose faults in the interaction between integrated units. Once all the modules have been unit tested, integration testing is performed.

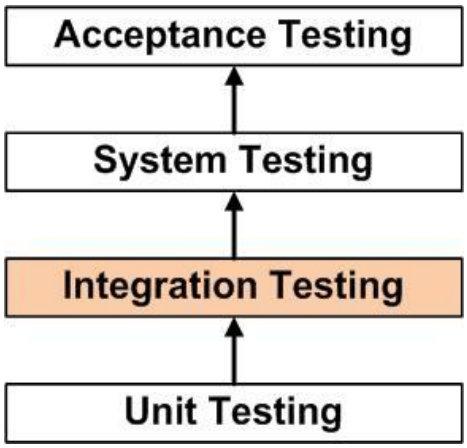


Fig 7.1.4

F. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. It increases the satisfaction of clients as they test application itself. The quality criteria of the software is defined in an early phase so that the tester has already decided the testing points.

1) Acceptance Testing For Data Synchronization

a) The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node.

b) The Route add operation is done only when there is a Route request in need.

c) The Status of Nodes information is done automatically in the Cache Updating process.

IX. CONCLUSION

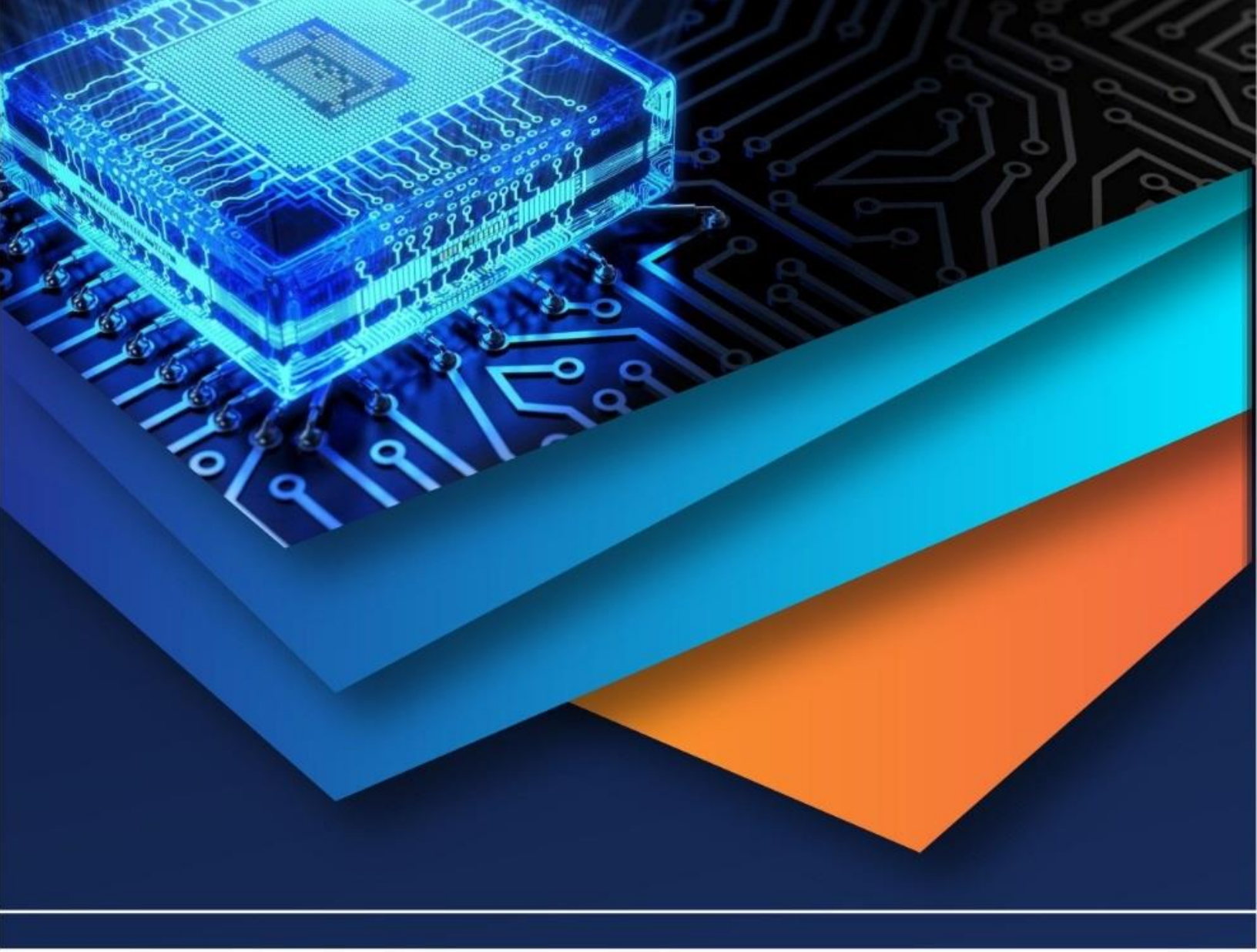
As a fragment of future work, we mean to endeavor different things with different likeliness limits regarding the blocked thing sets. Surely, one could utilize any likeliness work for a solitary item need. Unaided learning is the place you just have input information (X) and no comparing yield factors. The objective for unaided learning is to demonstrate the basic structure or circulation in the information so as to get familiar with the information during a fix. This is a fundamental fixing in our dynamic idea of information quality, and we give hypothetical confirmations to this property. As a major aspect of future work, we plan to try different things with various likeliness capacities for the prohibited item sets. To be sure, one could utilize any likeliness work for single object limitations. For whatever length of time that the impact of a fixed number of alterations on this likeliness capacity can be limited, at that point our methodology stays relevant for bigger classes of limitations. It is fascinating to perceive how to configuration fix calculations for standard limitations, for example, restrictive useful conditions, among others. Furthermore, the effect of client connection on the fixing procedure may bring about intriguing bits of knowledge.

X. FUTURE ENHANCEMENT

As a segment of future work, we intend to attempt various things with various likeliness capacities with respect to the precluded item sets. Certainly, one could use any likeliness work for a single object necessity. Unsupervised learning is where you only have input data (X) and no corresponding output variables. The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.

REFERENCES

[1] W. Fan and F. Geerts, Foundations of Data Quality Management, ser. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.
[2] I. F. Ilyas and X. Chu, "Trends in cleaning relational data: Consistency and deduplication," Foundations and Trends in Databases, vol. 5, no. 4, pp. 281–393, 2015.
[3] T. N. Herzog, F. J. Scheuren, and W. E. Winkler, Data Quality and Record Linkage Techniques. Springer, 2007.
[4] I. P. Fellegi and D. Holt, "A systematic approach to automatic edit and imputation," Journal of the American Statistical association, vol. 71, no. 353, pp. 17–35, 1976.
[5] X. Chu, I. F. Ilyas, and P. Papotti, "Holistic data cleaning: Putting violations into context," in ICDE, 2013, pp. 458–469.
[6] "Discovering denial constraints," PVLDB, vol. 6, no. 13, pp. 1498– 1509, 2013.
[7] P. Buneman, J. Cheney, W.-C. Tan, and S. Vansummeren, "Curated databases," in PODS, 2008, pp. 1–12.
[8] J. Rammelaere, F. Geerts, and B. Goethals, "Cleaning data with forbidden itemsets," in ICDE, 2017, pp. 897–908.
[9] W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis, "Conditional functional dependencies for capturing data inconsistencies," ACM TODS, vol. 33, no. 2, 2008.
[10] Z. Abedjan, X. Chu, D. Deng, R. C. Fernandez, I. F. Ilyas, M. Ouzzani, P. Papotti, M. Stonebraker, and N. Tang, "Detecting data errors: Where are we and what needs to be done?" PVLDB, vol. 9, no. 12, pp. 993–1004, 2016.
[11] W. Fan, F. Geerts, J. Li, and M. Xiong, "Discovering conditional functional dependencies," IEEE TKDE, vol. 23, no. 5, pp. 683–698, 2011.
[12] F. Chiang and R. J. Miller, "Discovering data quality rules," PVLDB, vol. 1, no. 1, pp. 1166–1177, 2008.
[13] X. Chu, I. F. Ilyas, P. Papotti, and Y. Ye, "Ruleminer: Data quality rules discovery," in ICDE, 2014, pp. 1222–1225.
[14] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur, "Dynamic itemset counting and implication rules for market basket data," in SIGMOD, 1997, pp. 255–264.
[15] G. Webb and J. Vreeken, "Efficient discovery of the most interesting associations," ACM TKDD, vol. 8, no. 3, pp. 1–31, 2014.
[16] P. C. Arocena, B. Glavic, G. Mecca, R. J. Miller, P. Papotti, and D. Santoro, "Messing up with bart: error generation for evaluating datacleaning algorithms," PVLDB, vol. 9, no. 2, pp. 36–47, 2015.
[17] J. Rammelaere and F. Geerts, "Revisiting conditional functional dependency discovery: Splitting the "C" from the "FD"," in ECML PKDD. Springer, 2018, p. TBD.
[18] J. Wang and N. Tang, "Dependable data repairing with fixing rules," JDIQ, vol. 8, no. 3-4, pp. 16:1–16:34, Jun. 2017.
[19] P. Bohannon, W. Fan, M. Flaster, and R. Rastogi, "A cost-based model and effective heuristic for repairing constraints by value modification," in SIGMOD, 2005, pp. 143–154.
[20] M. Dallachiesa, A. Ebaid, A. Eldawy, A. Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang, "Nadeef: a commodity data cleaning system," in SIGMOD, 2013, pp. 541–552.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY