*Journal of Advances in Science and Technology*
*Vol. 20, Issue No. 2, September-2023, ISSN 2230-9659*

# An Analysis on Attribute Selection and Token Formation used for Duplicate Record Detection

**Krishna Kant Tiwari[1]*, Dr. Qaim Mehdi Rizbi[2]**

[1] Research Scholar, Shri Krishna University, Chhatarpur, Madhya Pradesh, India

Email: kkit1984@gmail.com

[2] Associate Professor, Department of Computer Science & Application, Shri Krishna University, Chhatarpur, Madhya Pradesh, India

***Abstract-*** ***The data mining method relies heavily on data pre-processing. The data cleansing methods that work for some types of data may not work for others. Extensive experiments are conducted to analyze & assess a newly constructed method for attribute selection. The data cleaning processes involve reducing the amount of attributes to deal with noisy data & duplicate data. The experimental findings demonstrate that it is an extremely efficient and straightforward method for attribute selection by significantly reducing the attributes. Efficiently reducing the time required for subsequent data cleaning processes, such as token synthesis, record similarity, & deletion, is the primary goal of attribute selection for data cleaning. Smart tokens for data cleansing are formed using the token generation algorithm, which is appropriate for data that consists of numeric, alphabetic, & non-numerical elements. Duplicate data can be efficiently removed using token-based data cleaning. Attribute selection & token-based technique will both shorten the time required.***

***Keywords- Data, Duplicate Data, Attribute Selection, Token Formation, Algorithm, Quality***

----------X----------

## INTRODUCTION

There may be thousands of columns and millions of records in a data warehouse. With so much information stored in the data warehouse, cleaning it all up will be a daunting task. For instance, out of 50 columns in a dataset that indicate client characteristics, only 10 might be utilized for the purpose of identifying & detecting duplicates. Due to the massive volume of data, additional processing power and memory will be needed if unnecessary columns are removed during data cleaning. If you want to save time & effort on tasks like record similarity & deletion, attribute selection is a must. Dataset samples from customers are displayed in Table 1. It is unclear to users how many records there are, how many attributes there are, and how relative they are. When comparing two records, it is crucial to select the appropriate attributes (Kononenko 1997). All the subsequent phases build upon this initial step. When many attributes have the same name or when different names are used for the same attribute, it can lead to redundancies & inconsistencies within the attribute itself.

**Table 1: Sample records and attributes**

Attributes

| Records | Customer ID | Customer Name | Contact First Name | Contact Last Name | Contact Position | Address1 |
|---|---|---|---|---|---|---|
| | 1 | City Cyclists | Chris | Christianson | Sales Manager | 7464 South Kingsway |
| | 2 | Pathfinders | Christine | Manley | Sales Representative | 410 Eighth Avenue |
| | 3 | Bike-A-Holics Anonymous | Gary | Jannis | Sales Associate | 7429 Arbutus Boulevard |
| | 4 | Psycho-Cycle | Alexander | Mast | Sales Representative | 8287 Scott Road |
| | 5 | Sporting Wheels Inc. | Patrick | Reyess | Sales Associate | 480 Grant Way |
| | 6 | Rockshocks for Jocks | Heather | Davis | Sales Representative | 1984 Sydney Street |
| | 7 | Poser Cycles | Alex | Smith | Sales Agent | 8194 Peter Avenue |
| | 8 | Spokes 'N Wheels Ltd. | Kristina | Chester | Sales Representative | 3802 Georgia Court |
| | 9 | Trail Blazer's Place | Alexandra | Burris | Owner | 6938 Beach Street |
| | 10 | Rowdy Rims Company | Anthony | Shoemaker | Sales Agent | 4861 Second Road |

The attribute selection technique is shown sequentially in this flow diagram (Figure 1). In order to begin cleaning the data, the dataset must first be located. Attributes are examined in this dataset by classifying them according to their kinds, determining the relationships between them, and finally, choosing the most relevant characteristic based on its qualities. The attribute's data type, size, or length determine its type.

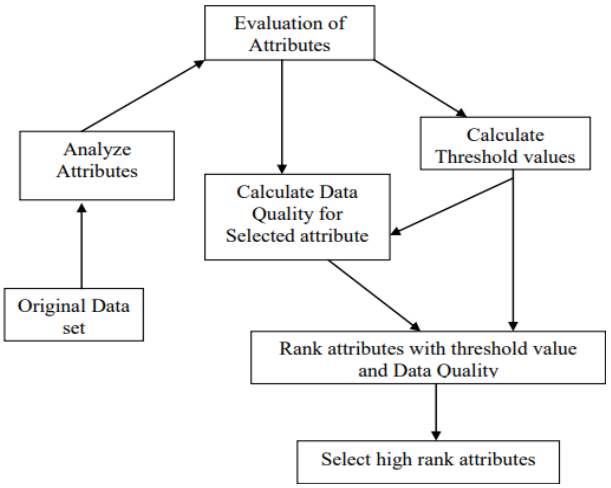


**Figure 1: Flow diagram of attribute selection**

The best qualities for cleaning the data are identified using the threshold value. High Threshold value, Data Quality, & High Rank are the three criteria used to measure the threshold value. In order to find the most powerful attributes for cleaning the data, a high threshold value is determined. Both the threshold & data quality values are used to rank the attributes. In order to expedite the data cleaning process, high-ranking attributes are subsequently chosen for the subsequent cleaning step. As part of the analysis, the user must specify the required attributes, their relationships, the data types, & total number of unique field values. The user must then give each of the chosen qualities a "weight" or "rank value" according to the data provided above. In order to streamline the data cleaning process and minimize user input, this research makes use of a software agent. As a last step in data cleansing, the attributes with the highest priority are chosen (Jiawei Han 2006). The practice of selecting the most desirable features in accordance with predetermined criteria is known as attribute selection. By eliminating superfluous or unnecessary attributes from the data warehouse, this attribute selection method speeds up & improves the accuracy of data cleansing. Key attribute identification, attribute classification based on high distinct value & low missing value, and measurement type of the attributes are the three criteria utilized for identifying appropriate attributes for the data cleaning process.

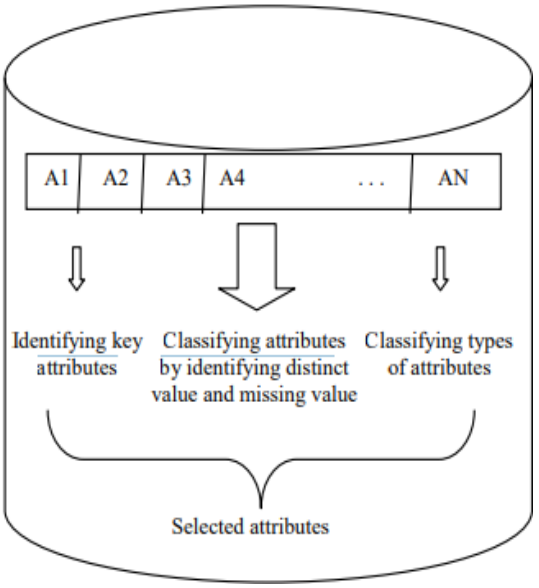


**Figure 2: Attribute selection using three main parameters**

The attribute selection with the parameters mentioned before is illustrated in Figure 2. Data warehouse calculations include the number of unique values, the percentage of attributes with missing values, and the kind of each attribute. In order to clean the data, the best attributes are chosen using these values.

a) Identifying key attributes

b) Classifying distinct and missing values

c) Classifying types of attributes

**Analysis of Attribute Selection Algorithm**

When cleaning data, an attribute selection algorithm will use the given constraints to choose which characteristics to clean. Figure 3 showcases the created algorithm for selecting attributes. As an initial step, the algorithm for selecting attributes chooses a relation schema R that contains N attributes. After that, it selects the relation model R's r table as an instance. Lastly, it chooses N characteristics from the relation schema R and stores them in the set Ai (A1,..., AN).

Input: N-Attributes, no. of tuples n, relation instance r

Output: S – Sub set of the attributes

Initialize: L - Temporary relation instance, x – Attribute set

Var: L - Temporary relation instance, x – Attribute set, F – Field set, i, j

begin

I. Analyze attribute set X to select best attributes
II. Calculate threshold value σ for each attribute
for each attribute xi, i<{ 0,1,2,…..N}
do
i) Assign threshold value as 1 for key attribute, put into L
ii) Calculate threshold σ with (σ: D ∧ M ∧ MT ∧ S)
a. Distinct (D) value of the attribute xi if tuple $\square_{i=1}^{n} t = t_i$
b. Missing (M) value of the attribute xi if tuple $\square_{i=1}^{n} ti = NULL$
c. Measurement types (MT) of attribute (ordinal, nominal, interval, and ratio) xi
Put into L.

end

III. Select attribute with high threshold value σ i, then put into L.
IV. Ignore attribute with low threshold value
V. Calculate data quality dqj of the selected attributes
for each attribute xi, i< {0,1,2,…..N}
for each field fj, j<{ 0,1,2,…..n}
compute

a. $Completeness_j = \frac{\sum_{i=0}^{N} completeness_{i,j}}{n}$
b. $Accuracy_j = \frac{\sum_{i=0}^{N} accuracy_{i,j}}{n}$
c. $Consistency_j = \frac{\sum_{i=0}^{N} consistency_{i,j}}{n}$
d. $dq_j = \frac{\propto * Completeness_j + \beta * Completeness_j + \gamma * Completeness_j}{n}$

end for
end for
VI. Rank attributes based on data quality and threshold value
VII. Select high rank attributes

End

**Figure 3: Algorithm for attribute selection**

A temporary relation schema L containing the name, type, missing value, distinct value, measurement type, & threshold value is obtained by this attribute selection technique. After that, it takes the name, type, and size of the attribute Ai from the relation schema R's relation instance r and assigns them to the temporary relation instance L. Determining the count of missing target values & distinct target values of the attribute Ai, as well as the percentage value, requires reading the tuples (records) from the selected relation instance r for each attribute. Instance L, a temporary relation, stores the fraction of items that are either absent or distinct. At last, for every attribute Ai, we find its measurement type and add it to the temporary relation instance L. For each target attribute Ai, we determine the threshold values by considering the measurement type, distinct values, & missing values. We then store these threshold values in the temporary relation instance L. Next, using the threshold values as a guide, determine the data quality power for a subset of the attributes S in the temporary relation L.

**Table 2: Attribute selection with sample data set**

| ColumnName | ColumnSize | DataType | Type Value | Uniques | Not Nulls | Truehole | Rank | Proceed |
|---|---|---|---|---|---|---|---|---|
| Customer ID | 4 | System.Int32 | 50 % | 100 % | 100 % | 0.8333 | 5 | ☑ |
| Customer Credit ID | 4 | System.Int32 | 50 % | 100 % | 100 % | 0.8333 | 4 | ☑ |
| Customer Name | 40 | System.String | 60 % | 84 % | 100 % | 0.8133 | 6 | ☑ |
| Contact First Name | 30 | System.String | 60 % | 81 % | 100 % | 0.8033 | 9 | ☑ |
| Contact Last Name | 30 | System.String | 60 % | 84 % | 100 % | 0.8133 | 8 | ☑ |
| Contact Title | 5 | System.String | 80 % | 5 % | 100 % | 0.6167 | 14 | ☐ |
| Contact Position | 30 | System.String | 60 % | 5 % | 100 % | 0.55 | 16 | ☐ |
| Last Years Sales | 8 | System.Decimal | 30 % | 76 % | 100 % | 0.6667 | 12 | ☐ |
| Address1 | 60 | System.String | 50 % | 84 % | 100 % | 0.78 | 10 | ☐ |
| Address2 | 20 | System.String | 75 % | 37 % | 47 % | 0.53 | 18 | ☐ |
| City | 20 | System.String | 75 % | 47 % | 100 % | 0.74 | 11 | ☐ |
| Region | 30 | System.String | 60 % | 37 % | 100 % | 0.6567 | 13 | ☐ |
| Country | 30 | System.String | 60 % | 17 % | 100 % | 0.59 | 15 | ☐ |
| Postal Code | 10 | System.String | 80 % | 82 % | 100 % | 0.8733 | 1 | ☑ |

To determine the threshold value, Table 2 displays the sample data set's attributes along with their corresponding missing values, distinct values, and data type. In order to get the ideal value for each characteristic, we compute the percentage of unique values, missing values, and measurement kinds. As shown in the graphic, in order to choose which attributes to clean next, high threshold value attributes are evaluated. Each attribute's variance in the threshold is displayed in Figure 3. Values between 0.9 and 1 are used to select the thresholds for the properties. Here, we use the aforementioned three parameters to determine the threshold values for every characteristic. The experimental outcomes dictate the variation of the threshold levels.

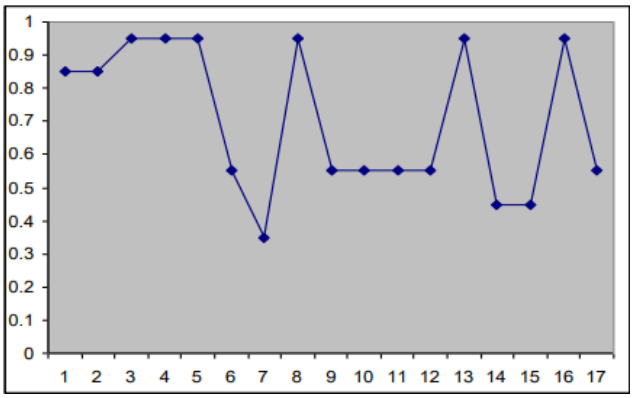


**Figure 4: Variance of threshold value for each attribute**

The last step in cleaning the data is to choose the high-quality qualities. The threshold values for the chosen properties are between 0.9 and 1. The following stage in cleaning is to choose the attribute contact information, which includes name, address, phone number, & postal code. When cleaning data with larger dimensionality (thousands of attributes), this attribute selection method proves to be efficient and successful. Any kind of data (nominal, numerical, etc.) can be used by the attribute selection process, which can also remove unnecessary or duplicate attributes. This attribute selection technique is also quite flexible and can easily process data with a wide variety of attributes. By following these guidelines, we can verify that the algorithm is of high quality. To efficiently minimize time & enhance performance for future data cleaning processes including token construction, record similarity, and deletion, attribute selection is used for data cleaning. The token-based method is just used for the attribute fields that have been explicitly chosen. Long string inputs to the similarity function necessitate a multi-pass method and increase comparison processing time. Token value is computed for both strings as DCAKU & compared, rather than comparing large strings like "Department of Computer Applications, Karunya University" with "Dept of Comp Appl, KU." One example is this. With the goal of speeding up data

cleansing and decreasing comparison time, the token based approach was devised.

**Assessing the Attributes' Quality**

In order to clean up data, one must first establish the quality of the attributes before selecting them. Attribute quality should mirror data cleansing effort. When trying to gauge an attribute's quality, there are two main methods:

i). Ignoring other attributes allows one to measure an attribute's quality (G. H. John 1994). Attribute selection aims, in part, to eliminate superfluous characteristics. Si = A − {Ai} denotes the set of selected attributes, where Ai is a relevant characteristic and A is the complete set of attributes.

ii). By choosing additional attributes with high threshold values (σ), one can assess an attribute's quality. Although some methods make use of indirect measures, the majority of procedures explicitly assign a quality metric to the property.

**TOKEN FORMATION**

Tokens are created for every attribute field that ranks highest. Before the token is formed, the following procedures are followed. Now, here are the steps:

**i) Remove unimportant tokens**

The initial stage of token formation involves eliminating unnecessary characters in order to obtain the most intelligent or optimal token for subsequent data cleaning. Special letters, ordinal forms, frequent words, stop words, title tokens, and greeting tokens are all examples of the insignificant tokens. Table 3 lists the common, insignificant tokens.

**Table 3: Unimportant characters**

| |
|---|
| **a. Special characters →** ` , ‘ ” < > - % + _ ( ) . * - $ # ! [ ] ^ \ @ : ; = ? \| { } ~ and etc<br>**b. Title or Salutation →** ‘Rev‘,’Dr’, ’Mr’, ‘Miss’, ‘Master’, ‘Madam’, ‘Sir’, ‘Chief’, ‘Ms’, ‘Mister’, ‘Shri’, ‘Drs’, ‘Dres instead of Dr.’, ‘Dr.’, ‘Mistress’, ‘Sis’, ‘Sri’, ‘Dear’, ‘Judge’, ‘Justice’, ‘Sister’<br>**c. Ordinal forms →** ‘st’, ‘nd’, ‘rd’, and ‘th’<br>**d. Common abbreviations →** ‘Pvt’, ‘Ltd’, ‘Co’, ‘Rd’, ‘St’, ‘Ave’, ‘Blk’, ‘Apt’, ‘Univ’, ‘Sch’, ‘Corp’ and etc<br>**e. Common words →** ‘and’, ‘the’, ‘of’, ‘it’, ‘as’, ‘may’, ‘than’, ‘an’, ‘a’, ‘off’, ‘to’, ‘be’, ‘or’, ‘not’, ‘I’, ‘about’, ‘are’, ‘at’, ‘by’, ‘bom’, ‘de’, ‘en’, ‘for’, ‘from’,’how’, ‘in’, ‘is’, ‘la’, ‘on’, ‘that’, ‘this’, ‘was’, ‘what’, ‘when’, ‘where’, ‘who’, ‘will’, ‘with’ and etc. |

**ii) Expand abbreviations using Reference table**

There are issues with token construction caused by the use of acronyms. In the process of creating tokens, the extension of acronyms is crucial. Table 4 provides a collection of commonly used acronyms. The log or reference table is where these acronyms are kept. The token construction relies on these tables as reference tables.

**Table 4: Reference Table with sample data**

| S. No. | Shortcut | Full form |
|---|---|---|
| 1 | Acc. a/c, A/C | account, account current |
| 2 | advt. | Advertisement |
| 3 | Apr. | April |
| 4 | Ave | Avenue |
| 5 | Co. | Company, country |
| 6 | Dept. | Department |
| 7 | Dep. | Departure |
| 8 | Est. | Established, estimated |
| 9 | Gov. | Government, governor |
| 10 | H.O | Head Office |
| 11 | Pvt | Private |
| 12 | Ltd | Limited |
| 13 | Rd | Road |
| 14 | Blk | Block |
| 15 | Apt | Apartment |

**iii) Formation of Tokens**

When it comes to creating tokens, various types of data are treated differently. The information could be in numerical, alphabetic, or both forms. The algorithm specifies the rules.

- **Numeric Tokens:** A variety of numerical data types can be formed using this approach, including phone numbers, social security numbers, street numbers, apartment numbers, and more. To begin, it changes the character to a number and then strips it of any unnecessary characters.
- **Alphabetic Tokens:** Contact names, client names, product names, book titles, etc., are ideal candidates for this rule of alphabetic token creation. It starts by removing unnecessary & stopping characters and expanding the abbreviations.
- **Alphanumeric Tokens:** For addresses, product codes, and the like, this rule for alphanumeric tokens works well. After sorting out the divided token, it divides alphanumeric into numeric and alphabetic groups before moving on to the next step.

```
Input: Tables with dirty data, Reference table, Selected attributes
Output: LOG table with tokens
Var: i, j, m – attribute set, n – no. of records
begin
For attribute i = 1 to m
for row j = 1 to n
do
   i)  remove unimportant characters
   ii) expand abbreviations using Reference table
   iii)if row(j) isnumeric then
      a. convert string into number
      b. sort or arrange the number in order
      c. form a token, then put into LOG table
   end if
   iv)if row(j) is aphanumeric then
      a. separate numeric and alphanumeric
      b. split alphanumeric into numeric and alphabetic
      c. sort numeric and alphabetic separately
      d. form a token, then put into LOG table
   end if
   v) if row(j) is alphabetic then
      a. select the first character from each word
      b. sort these letters in a specific order
      c. string them together
      d. if one word is present, take the first three character as token, then sort the characters
      e. form a token, then put into LOG table
   end if
end
```

**Figure 5: Algorithm for Token Formation**

A token creation algorithm is illustrated in Figure 5. The algorithm specifies rules for token formation. The algorithm's performance is dependent on the data type. Tokens are formed using the alphanumeric rule, for instance, when the address attribute is chosen. The tokens that are created are saved in the LOG table. Token keys for the address field are generated using Table 5. In this table, the rule for alphanumeric tokens is applied. The alphanumeric data is first separated into numeric & alphabetic parts, and then the alphabetic rule is applied. At last, it all comes together to form the token key.

**Table 5: Formation of Tokens for the address field**

| Customer Credit ID | Address | Token key |
|---|---|---|
| 1 | 7464 South Kingsway, Sterling Heights | 7464SKSH |
| 2 | 410 Eighth Avenue, DeKalb | 410EAD |
| 3 | 7429 Arbutus Boulevard, Blacklick | 7429ABB |
| 4 | 8287 Scott Road, Huntsville | 8287SRH |
| 5 | 480 Grant Way, San Diego | 480GWSD |
| 6 | 1984 Sydney Street, Austin | 1984SSA |
| 7 | 7655 Mayberry Crescent, Eden Prairie | 7655MCE |
| 8 | 413 Robson Lane, Winchester | 413RLW |
| 9 | 1842 St. Anne Place, Concord | 1842APC |
| 10 | 8404 Nelson Lane, Winchester | 8404NLW |

### iv) Maintaining LOG Table

In order to create a token for the chosen properties, the suggested algorithm is utilized. The LOG table is where the tokens that are formed are stored. Tokens representing the values entered into the specified attribute fields are temporarily stored in this LOG table. The LOG Table is now undergoing record comparison in order to detect duplicates. You may find the details of the sample LOG table with smart token in Table 6.

**Table 6: LOG Table with Smart Tokens**

| Customer Credit ID | Contact name key | Customer name key | Address key | Postal key |
|---|---|---|---|---|
| 1 | CC | CC | 7464SKSH | 48358 |
| 2 | CM | PA | 410EAD | 60148 |
| 3 | GJ | AABH | 7429ABB | 43005 |
| 4 | AM | PC | 8287SRH | 35818 |
| 5 | PR | ISW | 480GWSD | 92150 |
| 6 | DH | FJR | 1984SSA | 78770 |
| 7 | GW | HH | 7655MCE | 55327 |
| 8 | BM | CCGS | 413RLW | 22618 |
| 9 | PR | ACC | 1842APC | 01733 |
| 10 | CC | BCT | 8404NLW | 22616 |

Data cleansing smart tokens can be formed using the token construction algorithm, which works with all types of data types (numerical, alphabetic, and otherwise). Tokens that contain numbers, letters, & symbols are subject to three sets of regulations. One efficient outcome of token-based data cleansing is the elimination of duplicate data. Utilizing a token formation technique, tokens are created for values in certain attribute fields. The LOG stores these created tokens. Table 6. Compared to comparing tokens, comparing a full string is an expensive task. Consequently, defining the best and smartest token relies heavily on the token production process. Applying basic criteria for defining numeric, alphabetic, & alphanumeric tokens, this algorithm aims to define smart tokens from fields of selected numerous most significant properties. Smart token records, built from record field tokens, now make up the temporary table. The block-token-key algorithm is used to sort these smart token records. To increase data quality by lowering false-mismatches & actual mismatches, this study work generates block-token-key by considering more parameters. When creating block-token-keys, the parameters are:

- Power properties of high quality
- Plenty of data to identify duplicates
- Adequate values for the measurement types used

A sorted token table is the end product of this procedure; it is utilized to check for matches between nearby data. These tables are great for detecting duplicates. To facilitate the process of comparing recordings, the concept of "token records" was proposed. Token keys retrieved from

records are the sole means by which existing algorithms rank or block records. What follows is an explanation of the block-token-key.

## EXPERIMENTAL RESULTS

We use an attribute selection technique to pick the most relevant attributes that have sufficient data for finding duplicate records. Duplicate record identification is accomplished by using selected attributes. In order to detect duplicate data, the attributes that were chosen contain sufficient information. Various attribute values, numbers of duplicate records detected, & token creation are used to derive results in this chapter. Token formation and attribute selection algorithms are tested on the student dataset & Customer dataset, respectively. Choosing the right qualities is crucial for efficient duplicate detection and removal. One way to speed up cleaning planning is to utilize a token formation algorithm to create tokens based on the values of certain attribute fields.

### Attribute Selection with parameters

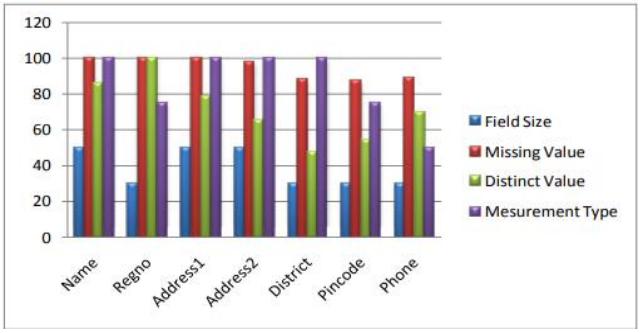


**Figure 6: Attribute selection in Student Dataset**

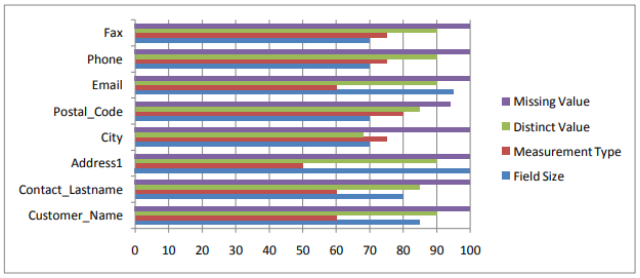


**Figure 7: Attribute selection in Customer Dataset**

Data cleaning performance & accuracy are impacted by the incorrect selection of attributes. Because of this, we make sure that the fields we choose for important data have enough information to spot record duplication. The data cleaning process's attribute selection is shown in Figures 6, 7. Field size, missing data, distinct values, and measurement type are the four main criteria used to determine the threshold value. When cleaning the data, the best qualities are chosen according to the threshold value. As a result of the name attribute's higher threshold value than the others, seven attributes will move on to the next data cleaning stage.

### Attribute Vs Duplicates

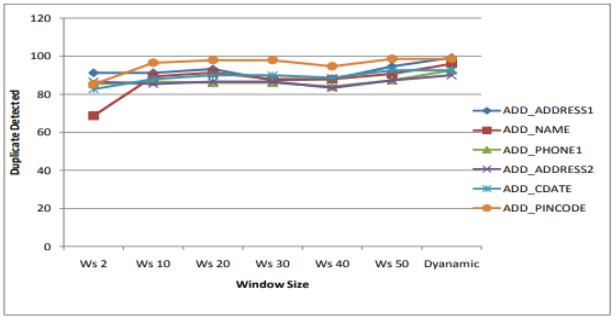


**Figure 8: Attribute Vs Duplicate detected with varying window size in Student Dataset**

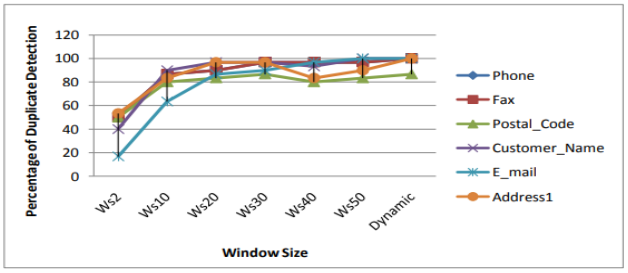


**Figure 9: Attribute Vs Duplicate detected with varying window size in Customer Dataset**

The selection of attributes & selection of window size are the major factors that determine identity of duplicates. The current approaches employ a fixed-size sliding window to reduce the amount of comparisons. Here, we employ field-value similarity to dynamically adjust the window size. When it comes to duplicate detection, our dynamic method yields the greatest results. For both static and dynamic window sizes, Figures 8, 9 demonstrate how the amount of detected duplicate records fluctuates with the slider window size. Different window sizes and dynamic sizes produce different duplicate detection results. Variation of attribute values for each execution setting window size (ranging from 10 to 50 and dynamic size) is used to evaluate this phenomenon.

### Duplicates Vs No. of Attributes

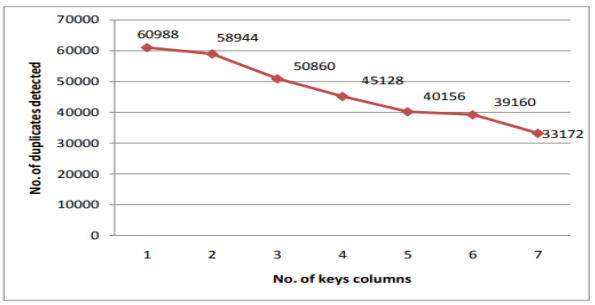


**Figure 10: Duplicate detected Vs No. of attribute selected in Student Dataset**

**Table 7: Key columns and No. of duplicate detected in Student Dataset**

| No. of Columns | Key Columns Selected | No. of duplicate detected |
|---|---|---|
| 1 | ADD_ADDRESS1 | 60988 |
| 2 | ADD_ADDRESS1 ADD_NAME | 58944 |
| 3 | ADD_ADDRESS1 ADD_NAME ADD_PHONE1 | 50860 |
| 4 | ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE | 45128 |
| 5 | ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL | 40156 |
| 6 | ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL ADD_PARENTTYPE | 39160 |
| 7 | ADD_ADDRESS1 ADD_NAME ADD_PHONE1 ADD_CDATE ADD_DEL ADD_PARENTTYPE ADD_PINCODE | 33172 |

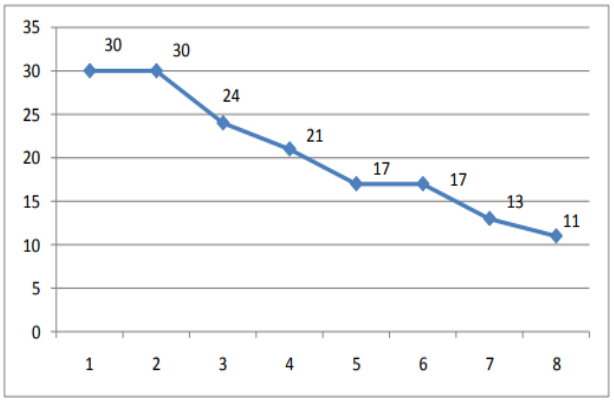


**Figure.11: Duplicate detected Vs No. of attribute selected in Customer Dataset**

**Table 8: Key columns and No. of duplicate detected in Customer Dataset**

| No. of Columns | Key Columns Selected | No. of duplicate detected |
|---|---|---|
| 1 | PHONE | 30 |
| 2 | PHONE FAX | 30 |
| 3 | PHONE FAX POSTAL_CODE | 24 |
| 4 | PHONE FAX POSTAL_CODE CUSTOMER_NAME | 21 |
| 5 | PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL | 17 |
| 6 | PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME | 17 |
| 7 | PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME CITY | 13 |
| 8 | PHONE FAX POSTAL_CODE CUSTOMER_NAME E_MAIL CONTACT_LAST_NAME CITY ADDRESS1 | 11 |

Attribute selection is the primary determinant of record matching algorithm efficiency. Figure 11 shows that the result is significantly affected by the choice of key column. The results of selecting key fields & number of approximate duplicate records found for those fields are displayed in Table 8. Finding exact and inexact duplicates will be more beneficial when numerous attributes are selected.

**Time Vs Token Formation**

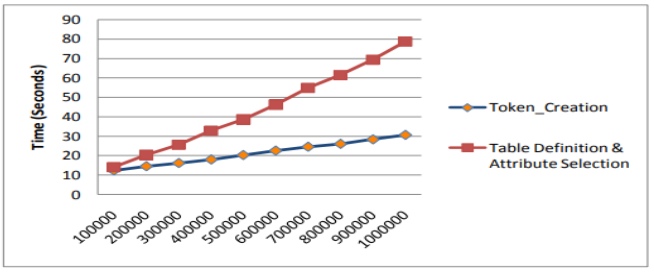


**Figure 12: Time taken Vs token formation, attribute selection with different data size**

In this study, the time required to form a token is minimal. It also takes a few seconds to choose the optimum attribute for data cleansing & load tables from the data warehouse (Figure 3.12). The duration varies according to the magnitude of the dataset.

## CONCLUSION

The usefulness & efficiency of this attribute selection strategy is demonstrated when dealing with larger dimensionality (thousands of characteristics) in the data cleaning process. Any kind of data (nominal, numerical, etc.) can be used by the attribute selection process, which can also remove unnecessary or duplicate attributes. This attribute selection technique is also very good at handling data with a wide variety of attribute kinds. By following these rules, we may be sure that the algorithm produces high-quality results. Efficiently reducing the time required for subsequent data cleaning processes, such as token synthesis, record similarity, and deletion, is the primary goal of attribute selection for data cleaning. One efficient outcome of token-based data cleansing is the elimination of duplicate data. A record in the LOG Table contains these created tokens. Compared to comparing tokens, comparing a complete string takes more time. In the subsequent data cleansing procedure, this token will serve as the blocking key. Accordingly, defining the most effective & smartest token relies heavily on the token production process.

## REFERENCES

1. Ali, A., Emran, N. A., Asmai, S. A., & Thabet, A. (2018). Duplicates detection within incomplete data sets using blocking and dynamic sorting key methods. International Journal of Advanced Computer Science and Applications, 9(9).

2. Bilenko, M., Mooney, R.J.: Adaptive Duplicate Detection Using Learnable String Similarity Measures, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03), Washington, DC, August 2003

3. Chen Shengxin, Intelligent Data Warehousing: From Data Preparation to Data Mining, Language: ENGLISH. 242p. 16x24 Hardback, Publication date: 01-2002.

4. Elgamal, F., Mosa, N. A., & Amasha, N. A. (2014). Application of framework for data cleaning to handle noisy data in cloud computing. International Journal of Soft Computing and Engineering, 3, 226-231.

5. F. Naumann and M. Herschel, "An introduction to duplicate detection," Synthesis Lectures on Data Management, vol. 2, no. 1, pp. 1–87,2010.

6. Kaur, R., Chana, I., & Bhattacharya, J. (2018). Data deduplication techniques for efficient cloud storage management: a systematic review. The Journal of Supercomputing, 74, 2035-2085.

7. Leesakul, W., Townend, P., & Xu, J. (2014, April). Dynamic data deduplication in cloud storage. In 2014 IEEE 8th International Symposium on Service Oriented System Engineering (pp. 320-325). IEEE.

8. Patil, R. Y., & Kulkarni, R. V. (2012). A review of data cleaning algorithms for cloud computing systems. International Journal of Computer Science and Information Technologies, 3(5), 5212-5214.

9. Rajakumari, K. E. (2019, February). Comparison of Token-Based Code Clone Method with Pattern Mining Technique and Traditional String Matching Algorithms In-terms of Software Reuse. In 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) (pp. 1-6). IEEE.

10. Reddy, S. L., & Prasad, K. R. (2019) Study on advantages of deduplication in cloud computing. Journal of Engineering Sciences. Vol 10,Issue3, MARCH/2019 ISSN NO:0377-9254

11. Selvi, S. A. E., & Anbuselvi, R. (2015, March). An Analysis of Data Replication Issues and Strategies on Cloud Storage System. In International Journal of Engineering Research & Technology (IJERT), NCICN-2015 Conference Proceedings, pp18-21.

12. Zafar, F., Khan, A., Malik, S. U. R., Ahmed, M., Anjum, A., Khan, M. I., ... & Jamil, F. (2017). A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends. Computers & Security, 65, 29-49.

**Corresponding Author**

**Krishna Kant Tiwari***

Research Scholar, Shri Krishna University, Chhatarpur, Madhya Pradesh, India

Email: kkit1984@gmail.com