Data Quality Assessment

ZIMING SONG, New York University, USA
TIANYUE HUANG, New York University, USA
KEYU PANG, New York University, USA

In the age of big data, data has emerged as a critical resource for data-driven decision-making across diverse fields. However, the quality of data is paramount for its effective utilization. Without robust quality assurance measures, the reliability of analyses conducted on such data is compromised, potentially leading to biased or invalid conclusions. This paper proposes methodologies to categorize null, misspelling, abbreviation and anomalies in open datasets, aiming to enhance their reliability and utility for informed decision-making. We implement our approach on two different dataset and get good recall and precise.

Additional Key Words and Phrases: Data Quality, Data Cleaning

ACM Reference Format:
Ziming Song, Tianyue Huang, and Keyu Pang. 2024. Data Quality Assessment. 1, 1 (May 2024), 13 pages. https://github.com/Iris-Song/PerHapS

1 INTRODUCTION

In today's era of big data, open data has become an indispensable data source for data-driven decision-making. Open data presents a wealth of information for diverse applications ranging from decision-making to academic research. However, it is imprecise to state that the proliferation of data propels the development of major data-driven applications since the utility of data is heavily contingent on its quality. Data plagued with issues such as incorrect values, inconsistencies, and missing information can lead to erroneous conclusions, undermining the very purpose of data analytics.

Open data is often generated from a multitude of sources, each with varying standards of data entry and maintenance. This diversity inherently introduces discrepancies and errors. For instance, a dataset might contain incomplete or outdated information, mislabelled or misspelled entries, and inconsistent formats, all of which compromise its integrity. Without a robust mechanism to ensure data quality, the reliability of analyses conducted by these users can be severely impacted. Poor data quality can lead to biased or invalid models, which in turn can perpetuate errors and misconceptions. In fields such as machine learning and artificial intelligence, where data is the backbone of algorithmic decision-making, the repercussions of low-quality data can be particularly significant.

Thus, the development of methodologies to identify and categorize anomalies in datasets is not just a technical necessity but a foundational aspect of ensuring that open data serves its intended purpose effectively. By classifying data into valid, misspelled, invalid, and null categories, and addressing the unique challenges inherent in open data quality assessment, this project aims to enhance the reliability and usefulness of these datasets, facilitating better, more informed decisions across various domains.

Authors' addresses: Ziming Song, New York University, Brooklyn, New York, USA, zs2815@nyu.edu; Tianyue Huang, New York University, Brooklyn, New York, USA, th3113@nyu.edu; Keyu Pang, New York University, Brooklyn, New York, USA, kp2344@nyu.edu.

Manuscript submitted to ACM

2 RELATED WORK

The field of data quality assessment has been a subject of extensive research over the past few years, with several studies contributing valuable insights and advancements. With respect to different categories of suspicious/anomalous values in table, different ideas were proposed to identify or tackle the problem.

2.1 NULL value

The fundamental causes and implications were thoroughly studied. DMV (Disguised Missing Values) can result from deliberate data manipulation and fraud, electronic data entry systems, data coding practices, etc. various strategies were explored to identify DMV, such as distributional analysis, outlier detection and suspicious value analysis [15]. However, these methods heavily relays on domain-specific knowledge. [8] leveraged the EUS heuristic, which operates on the principle that the projected database of a disguise value contains a large unbiased sample of the whole dataset. This idea helps to unmask the disguised missing value even without domain background knowledge. Building upon [8], FAHES[17] utilized mutual information and other techniques to further improve the time efficiency and detection effectiveness.

2.2 Misspelling

The identification and correction of misspellings in textual data has been a long-standing challenge in natural language processing (NLP). Misspellings can arise from a variety of sources, including typographical errors, phonetic mistakes, and the lack of standardized spelling in some languages. Ensuring the accuracy of written content is paramount in NLP and text processing field.

Recent advancements have focused on machine learning and statistical approaches for misspelling detection. [10] provides an early comprehensive review of methods for automatic misspelling detection and correction, ranging from rule-based approaches to more complex probabilistic models.

Our main method integrates machine learning techniques with a comprehensive dictionary derived from the GloVe (Global Vectors for Word Representation) model.

2.3 Abbreviation

In the rapidly evolving of digital communication, abbreviations and acronyms have become ubiquitous. However, the misuse of abbreviations can lead to misunderstandings.

Initial studies in abbreviation checking were largely focused on specific fields, such as medicine and academia, based on predefined dictionaries and rule-based algorithms. Recent research has leveraged machine learning techniques to detect the accuracy of abbreviation, such as decision tress, support vector machines, and neural networks.

2.4 Invalid Value

In dataset, invalid values often manifest in several forms, including irrelevant values, outliers, or inconsistent data types. Particularly, recent research on invalid values focuses on outlier detection. Clustering algorithms like CLARANS[14], DBSCAN[5], BIRCH[19] and CURE[7] consider outliers, but only to the point of ensuring that they do not interfere with the clustering process. Further, the definition of outliers used is in a sense subjective and related to the clusters that are detected by these algorithms. Thus, [1] addresses the problem of detecting deviations – after seeing a series of similar data, an element disturbing the series is considered an exception. [18] propose a novel formulation for distance-based

outliers that is based on the distance of a point from its k^{th} nearest neighbor. The authors rank each point on the basis of its distance to its k^{th} nearest neighbor and declare the top n points in this ranking to be outliers.

KNN[2] employs the k-nearest neighbors approach, outlier anomalies based on the largest distances(or median distances, mean distances)to their nearest neighbors. Additionally, [13] explore the intricacies of Isolation Forest (IForest), a cutting-edge tree-based ensemble approach that excels in isolating anomalies by randomly partitioning data points. Angle-Based Outlier Detection (ABOD)[9] emerges as another formidable contender in our analysis, leveraging geometric insights by measuring the angles between data points to identify outliers with precision. Meanwhile, the Histogram-based Outlier Score (HBOS)[6] method harnesses the power of histograms to model the underlying data distribution efficiently, offering a novel perspective on anomaly detection. The Ensemble of Cluster-based Outlier Detection (ECOD)[12] deploys multiple clusterings to unearth outliers from varying perspectives, enhancing detection robustness. Feature Bagging[11], on the other hand, generates multiple models on subsets of features and aggregates their results to enhance anomaly detection robustness.

3 PROBLEM FORMULATION

We develop an automated data quality assessment tool to help to provide users with a review about data quality, in the form of classifying column values into four different categories: valid value, null value, misspelling/abbreviation of a valid value, invalid value,. As detecting valid value is trivial, we mainly focus on how to detect Misspelling/Abbreviation of a valid value, invalid value and null value. The rest of the values are valid values.

NULL Value: This represents missing or unknown data. NULL values can be explicit (like "NULL" or "N/A") or implicit, disguised as seemingly valid data (such as "999-999-999" for phone numbers). Identifying NULL values, especially the implicit ones, is essential for accurate data analysis and can involve nuanced detection methods beyond standard null checks.

Misspelling/Abbreviation of a Valid Value: This category includes values that are essentially correct but may contain typos or are presented as abbreviations. For example, "Nwe York" instead of "New York," or "NY" as an abbreviation for "New York." The key task here is to recognize these variations and correct or expand them to their standard forms, maintaining consistency in the dataset.

Invalid Value: Invalid values are entries that do not fit the expected format, type, or logical range of the dataset. This could range from alphanumeric strings in a purely numeric field to implausible or out-of-range data (like a negative age). These values are often more than simple errors or variations and might indicate deeper issues with data collection or entry.

4 METHODS, ARCHITECTURE AND DESIGN

4.1 Dataset

In this project, we use two datasets in NYC Open Data. NYC Open Data is free public data published by New York City agencies and other partners. One dataset we use is [3]. The other is [4]. Both datasets are used in NULL value and invalid value detection. Misspelling and abbreviation detection only use [3].

4.2 Methods

To ensure quality of data, it is important to find null values, misspelling and abbreviation value of valid values and invalid values.

Detecting explicit null values is trivial. Many libraries such as "Pandas" provide explicit null checker to identify null values. However, detecting disguised null values undergoes much more complex process. Firstly, We used "Pandas" isNull() to find explicit null values in the dataset. We also applied regular expression to find entries with null indicators such as "unknown" or "missing". For those disguised null values without explicit null indicators, we decided to use FAHES [17] to unmask DMVs that replace MAR/MCAR (Missing At Random/ Missing Completely At Random) values.Under the MCAR or MAR models, detecting the values that replace the missing values and used frequently could be achieved by removing one of the frequent values in a given attribute and testing if the resulting missing cells follow either models.If the resulting empty cells follow either models then that value is likely to be a DMV. This process could be repeated over the most frequent values in each attribute and the values that satisfy the MCAR/MAR assumption are reported as DMV candidates.[17] Moreover, a column conforming to a repetitive pattern (e.g., 999-999-999, 123456789) may also suggest potential DMVs. FAHES [17] possesses such functionality, though it is integrated within other detection tools. Therefore we isolated the repetitive pattern discovery logic and made minor modifications so that it better suit the requirements of our project while utilizing the original pattern discovery structure. The idea is to calculate the standard deviation of character differences in a string to analyze patterns of change or repetition in the characters. To prevent large character differences between char and number and to ensure correct detection of such pattern in phone number or date, non-numerical values are eliminated if numerical values are prevalent.

Misspelling checking is an important research area in natural language processing (NLP). Checking using GloVe (Global Vectors for Word Representation) is a method based on semantic similarity. GloVe is a pre-trained word vector model that captures co-occurrence relationships between words and encodes these relationships into dense vector representations by training on a large corpus. This representation makes semantically similar or related words close to each other in vector space. We utilize spellcheck python model to label. If one row contain multiple words, we individually check each work within the value and any detection of misspellings results in marking this value to be 'misspell', otherwise 'valid'. After labelling, we utilize the GloVe model to predict misspelling, comparing those two results to calculate recall and precision.

As for abbreviation, we implement a rule-based matching algorithm that is if a word begins with a letter, we preserve only the initial letter of each word and convert it to uppercase, if a word begins with a numeral, we retain only the numeric part. For example, the phrase '26th West' would be abbreviated as '26W'. We apply this rule to assess the accuracy of the full names and their corresponding abbreviations in our table.

As for invalid value detection, according to different data type, we use different methods. Firstly, according to invalid type cases, like finding invalid date format, we use regular expression to check. For numeric data, invalid data may fall in unreasonable range or contain alpha characters. So, we check if those numeric data in non-numeric format. Secondly, for semantic invalid, like finding invalid country, we use python library pycountry[16] to find irrelevant value. For example, 'OTHER ASIAN' is not a valid country. `countries.lookup` in [16] can be used to check if an input string is a valid country name. Thirdly, we use eight different algorithms to find outliers, checking if the number is in an acceptable range. For example, a person's age cannot less than 0. The eight algorithms are. KNN(using largest, mean, median distance respectively)[2], Isolation Forest[13], ABOD[9], HBOS[6], ECOD[12], Feature Bagging[11]. To implement the eight outlier detection algorithm, we use pyod[20] library. We compare the eight algorithm on [3][4] dataset on one dimension and two dimension respectively. If a data point is detected by any of the eight algorithm as outlier, the data point is considered as outlier.

4.3 Architecture

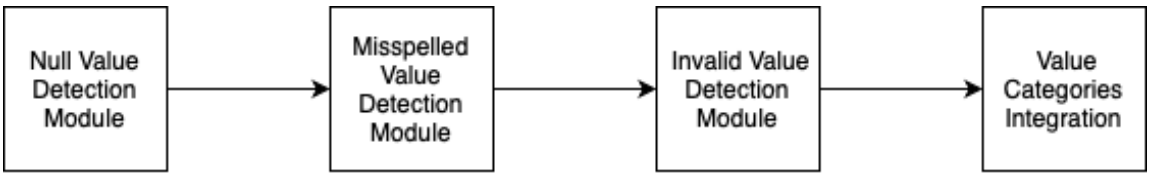

Fig. 1. Project Architecture

For the architecture of a project aimed at developing techniques to classify column values, a multi-layered and modular approach would be effective. The original dataset will be loaded to go through each process of null value, misspelled value and in value value detection. In the final value categories integration part, we tackle possible dual tagging and categorize the rest of the column values as valid.

5 EXPERIMENT AND RESULTS

Recall and precision result are calculated on used dataset. Since we could not find the labeled dataset, we labeling [3][4] manually assisted with ChatGPT 4. The labeled datasets are used as the ground truth in evaluation.

5.1 Recall

Each of the type is calculated respectively. The result of recall are shown in Table 1. The formula of recall is as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives+False Negatives}}$$

Table 1. Recall

Dataset	Type	Recall
[3]	Null	99.99%
[3]	Misspelling	47.71%
[3]	Invalid	88.30%
[4]	Null	92.43%
[4]	Invalid	83.61%
[4]	Abbreviation	100%

5.2 Precision

Similar to recall, each of the type is calculated respectively. The result of recall are shown in Table 2. The formula of recall is as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives+False Positives}}$$

Table 2. Precision

Dataset	Type	Precision
[3]	Null	99.98%
[3]	Misspelling	98.32%
[3]	Invalid	70.94%
[4]	Null	100%
[4]	Abbreviation	85.71%
[4]	Invalid	71.83%

5.3 Data Visualization

The Log-Scale missing values distribution is shown in Fig.2 and Fig.. In 2, "-1" and "UNKNOWN OR NOT STATED" was detected by FAHES [17] , Kumkum was identified as a repetitive term and other columns were identified by regular expressions. In 3, the missing column values were all detected by Pandas isNull() function.

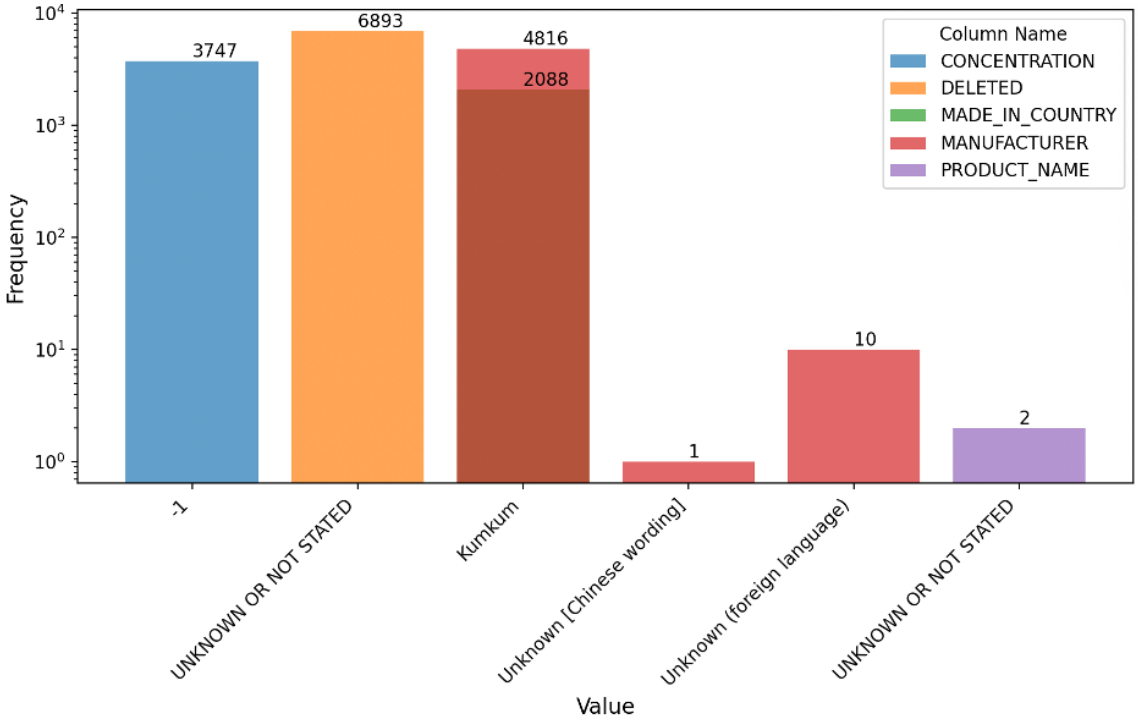

Fig. 2. Log-Scale NULL Distribution For [3]

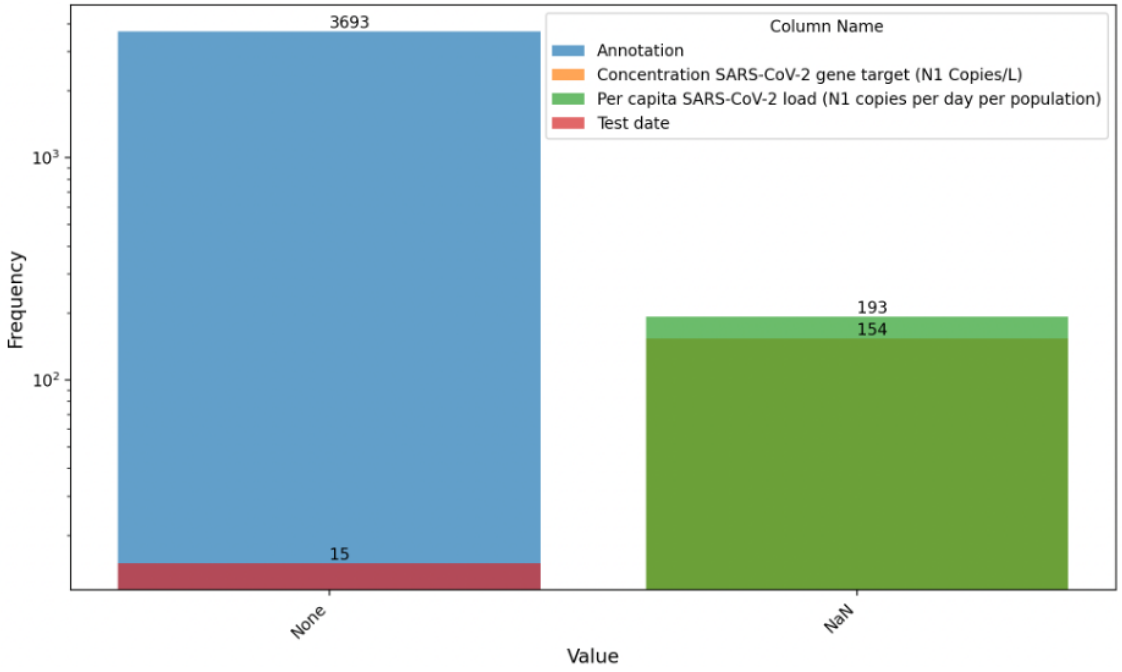

Fig. 3. Log-Scale NULL Distribution For [4]

As for detecting misspellings, we used regular expressions to transform the text value to remove symbols and digits. After that, we performed misspelling detection. We found Top 10 common spelling error words in 'MANUFACTURER' and Top 10 corrections Fig.4.

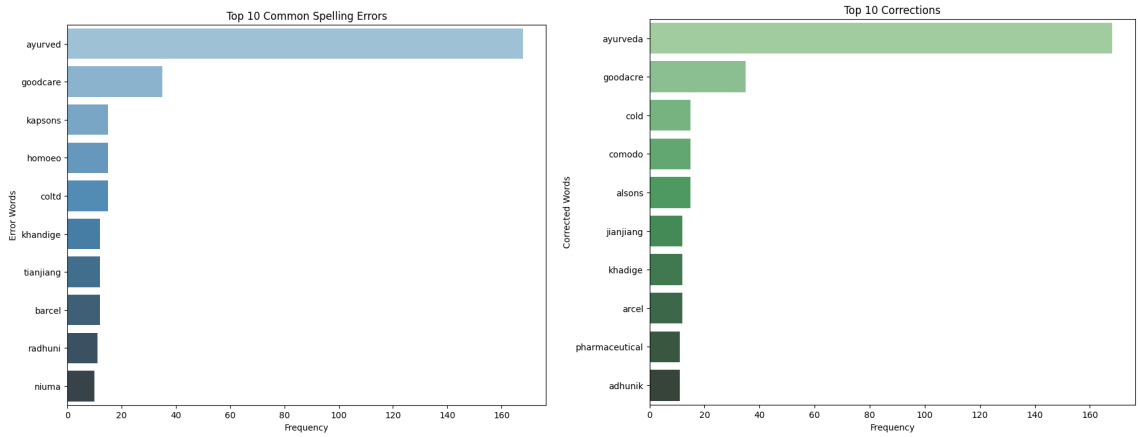

Fig. 4. Top 10 Common Spelling & Corrections

As for detecting invalid value, on dataset[4], we use KNN(using largest, mean, median distance respectively)[2], Isolation Forest[13], ABOD[9], HBOS[6], ECOD[12], Feature Bagging[11] as methods to find outliers. We apply these

eight algorithm on numeric column using one dimension and two dimension and visualize the result respectively. The results of one normalized dimension of column "Concentration SARS-CoV-2 gene target (N1 Copies/L) ", "Per capita SARS-CoV-2 load (N1 copies per day per population)", "Population Served, estimated " are respectively shown in Fig.8, Fig.9, Fig.7. X-axis is the normalized one dimension value. Also, we apply the algorithm on two dimension, and the data is also normalized. The results are shown in Fig.10, Fig.11, Fig.12,

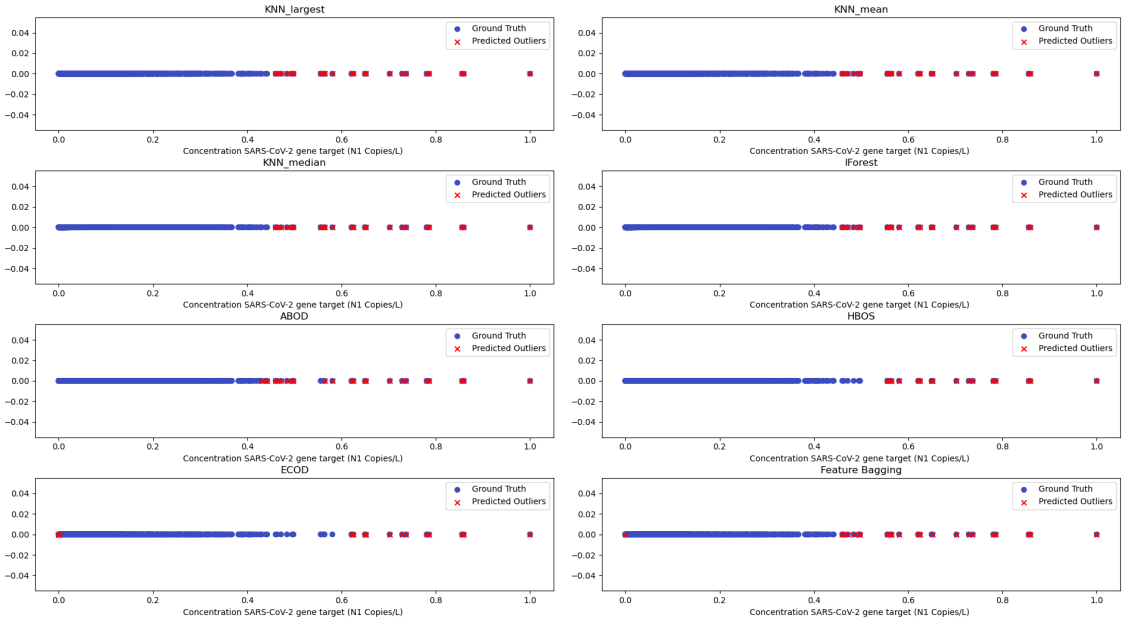

Fig. 5. Outlier Detection of "Concentration SARS-CoV-2 gene target (N1 Copies/L)"

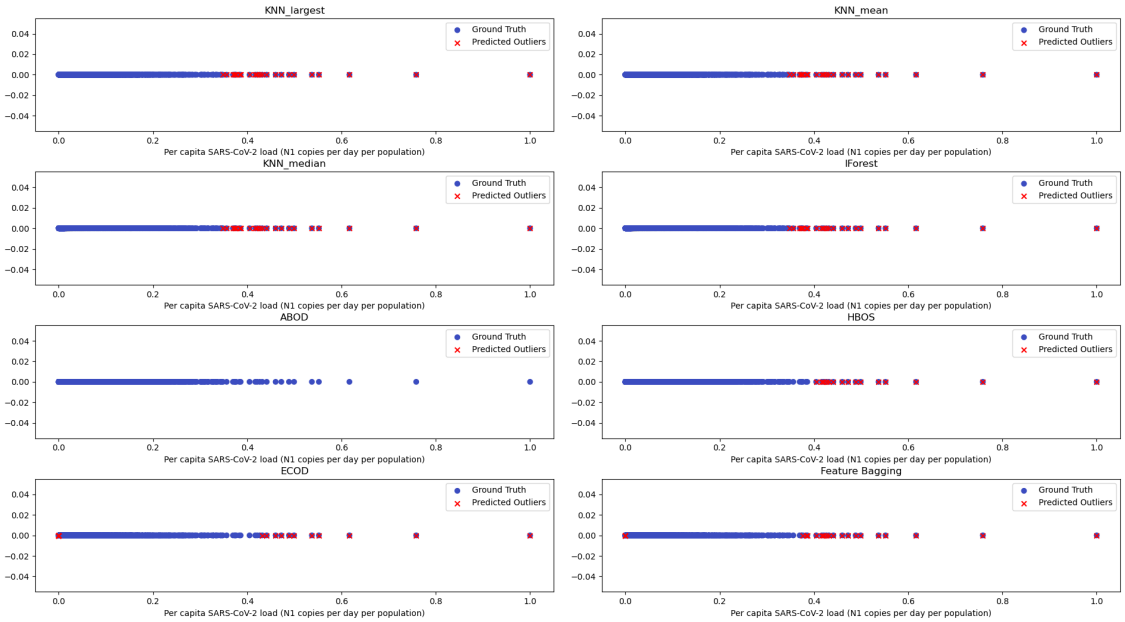

Fig. 6. Outlier Detection of "Per capita SARS-CoV-2 load (N1 copies per day per population)"

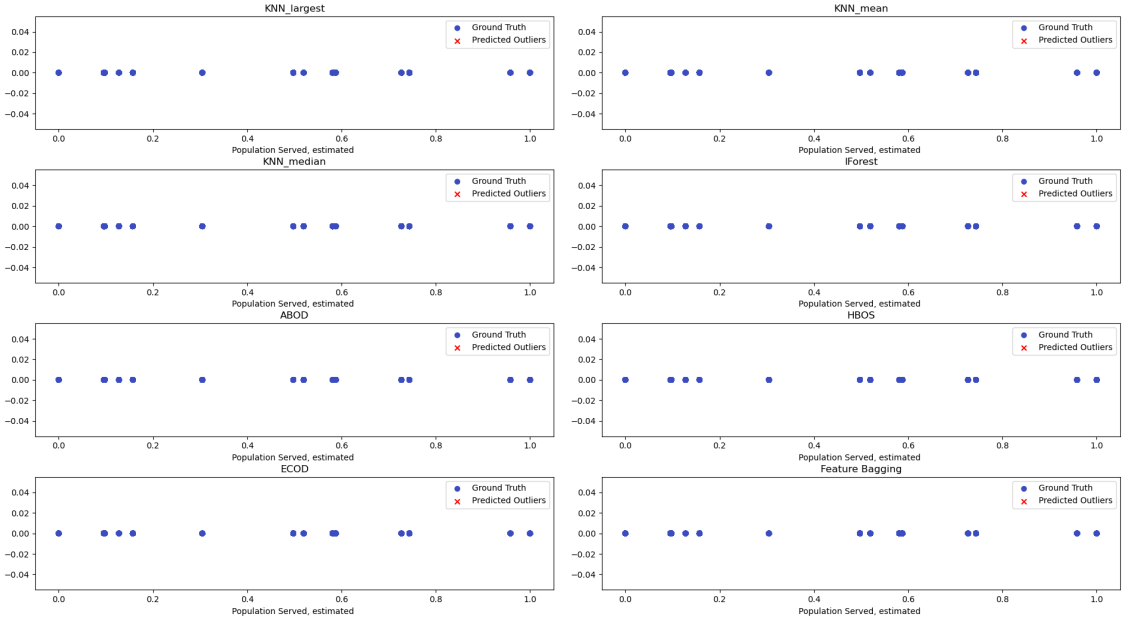

Fig. 7. Outlier Detection of "Population Served, estimated "

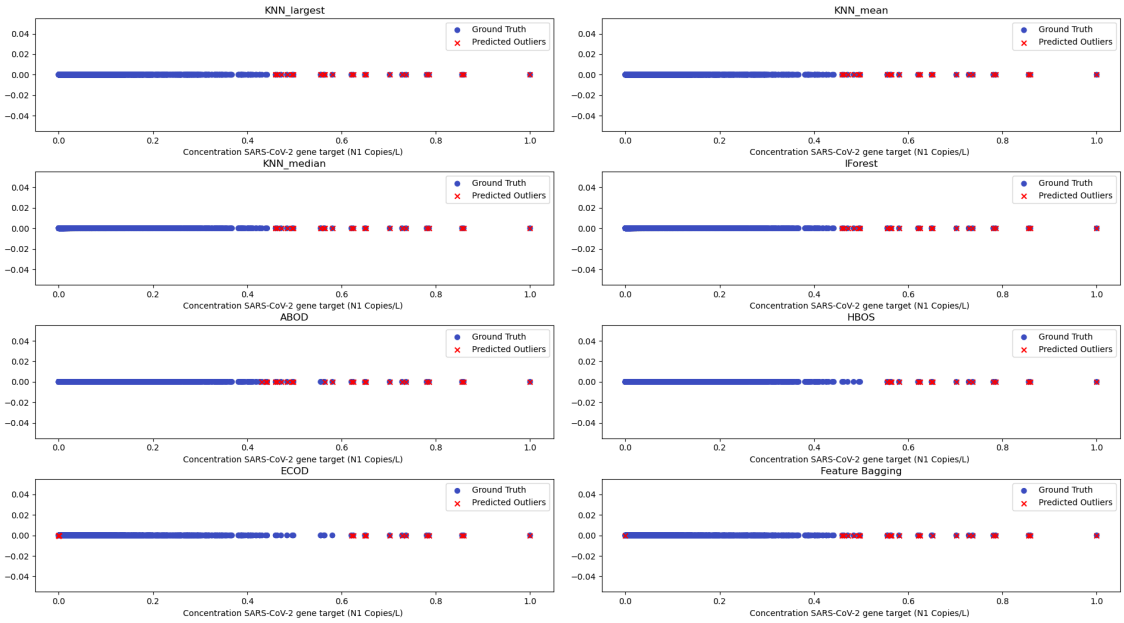

Fig. 8. Outlier Detection of "Concentration SARS-CoV-2 gene target (N1 Copies/L)"

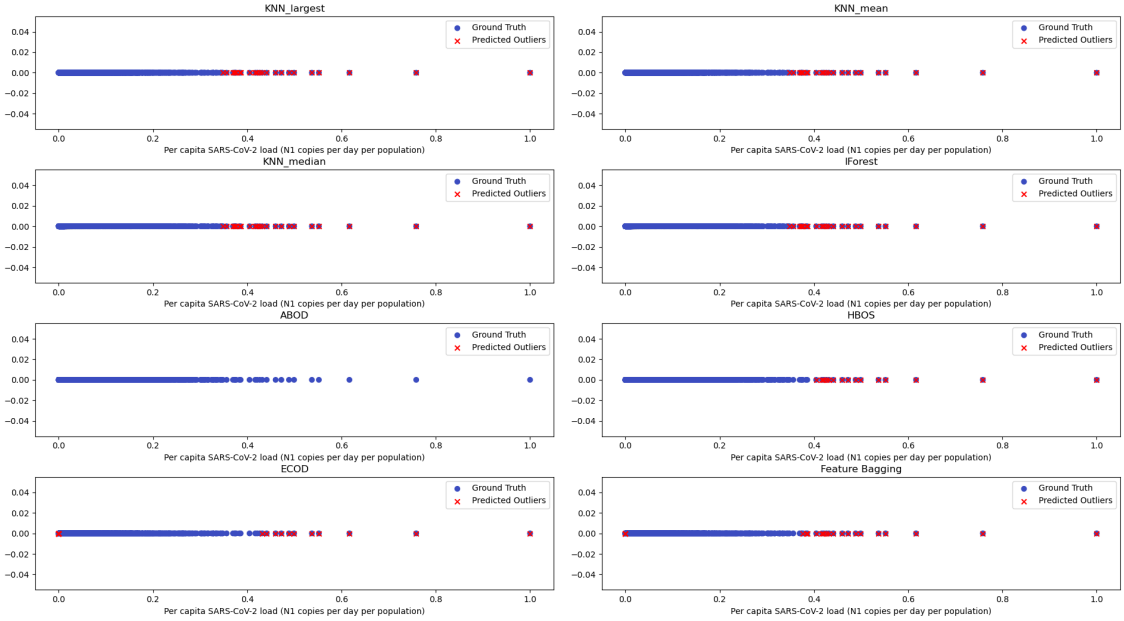

Fig. 9. Outlier Detection of "Per capita SARS-CoV-2 load (N1 copies per day per population)"

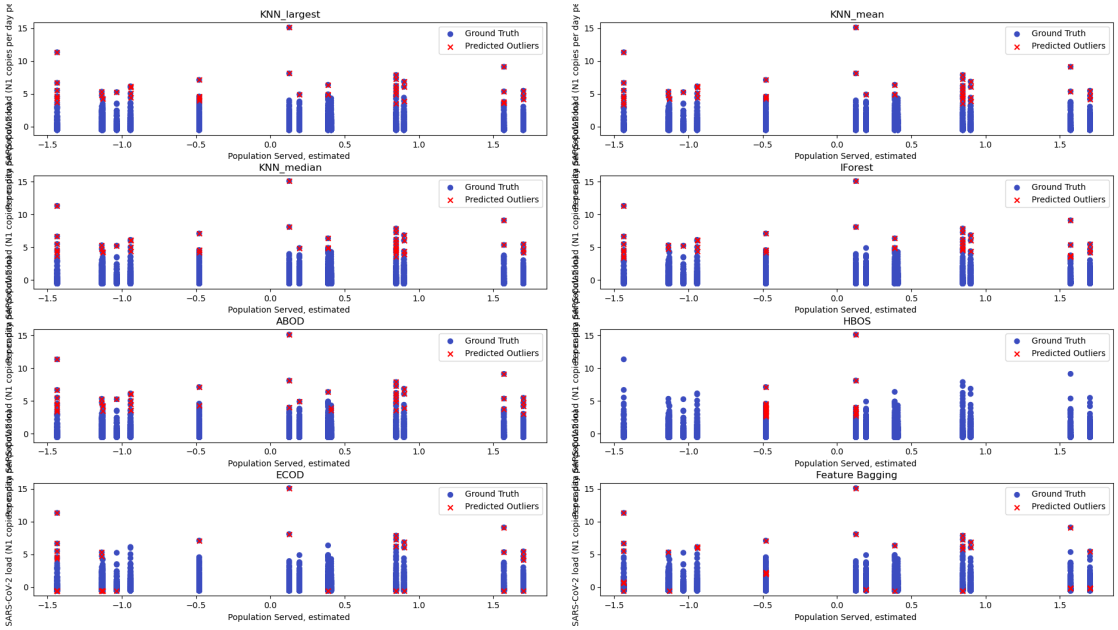

Fig. 10. Outlier Detection of two dimension

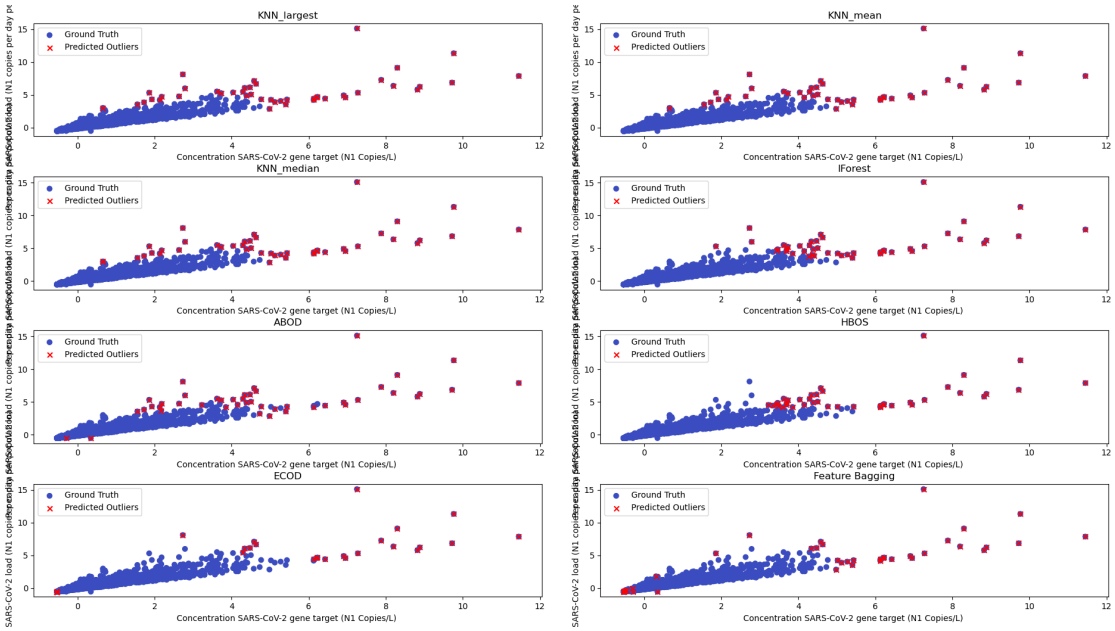

Fig. 11. Outlier Detection of two dimension

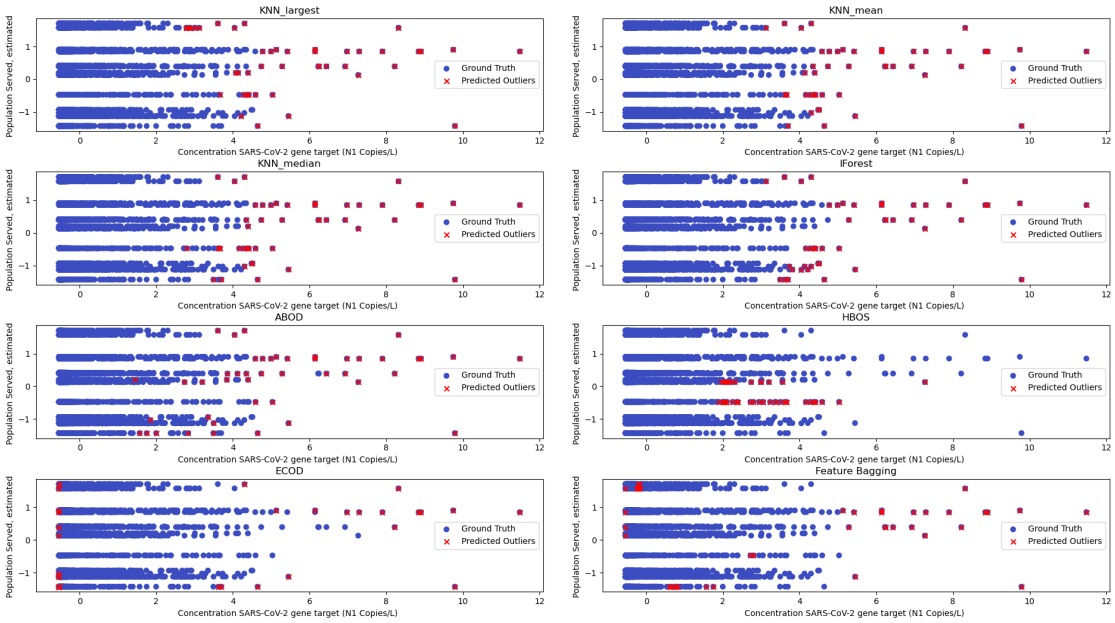

Fig. 12. Outlier Detection of two dimension

5.4 Limitation

Although our method achieves good performance on the dataset, it also has some limitations:

1. Disguised null detection might fail on MNAR (Missing Not At Random) column values.

2. Disguised null detection might fail when the missing values don't contain explicit null indicators and are not frequently used.

3. Some method like regular expression apply on specific columns, and these columns are selected manually and some regular expressions are derived from observation.

4. Our current method for checking abbreviations has limitations as it relies only on specific rules, and abbreviations can evolve to have increasingly diverse meanings over time.

5. To improve recall, we could explore using alternative algorithms that may offer better performance in identifying misspelled words.

6. Outlier detection method only applies on numeric data type.

REFERENCES

[1] Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. 1996. A Linear Method for Deviation Detection in Large Databases. In *Knowledge Discovery and Data Mining*. https://api.semanticscholar.org/CorpusID:15564115

[2] T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13, 1 (1967), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[3] NYC Open Data. 2018. *Metal Content of Consumer Products Tested by the NYC Health Department.* Retrieved February 7, 2024 from https://data.cityofnewyork.us/Health/Metal-Content-of-Consumer-Products-Tested-by-the-N/da9u-wz3r/about_data

[4] NYC Open Data. 2022. *ARS-CoV-2 concentrations measured in NYC Wastewater.* Retrieved March 13, 2024 from https://data.cityofnewyork.us/Health/SARS-CoV-2-concentrations-measured-in-NYC-Wastewat/f7dc-2q9f/about_data

[5] Dingsheng Deng. 2020. DBSCAN Clustering Algorithm Based on Density. In *2020 7th International Forum on Electrical Engineering and Automation (IFEEA)*. 949–953. https://doi.org/10.1109/IFEEA51475.2020.00199

[6] Markus Goldstein and Andreas Dengel. 2012. Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm.

[7] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. 1998. CURE: an efficient clustering algorithm for large databases. *SIGMOD Rec.* 27, 2 (jun 1998), 73–84. https://doi.org/10.1145/276305.276312

[8] Ming Hua and Jian Pei. 2007. Cleaning Disguised Missing Data: A Heuristic Approach. In *Proceedings of the KDD '07 Conference.* https://dl.acm.org/doi/10.1145/1281192.1281294

[9] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. 2008. Angle-based outlier detection in high-dimensional data. In *Knowledge Discovery and Data Mining.* https://api.semanticscholar.org/CorpusID:3072058

[10] K. Kukich. 1992. Techniques for Automatically Correcting Words in Text. *Comput. Surveys* 24 (1992), 377–439. https://doi.org/10.1145/146370.146380

[11] Aleksandar Lazarevic and Vipin Kumar. 2005. Feature bagging for outlier detection. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (Chicago, Illinois, USA) *(KDD '05).* Association for Computing Machinery, New York, NY, USA, 157–166. https://doi.org/10.1145/1081870.1081891

[12] Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H. Chen. 2022. ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions. *IEEE Transactions on Knowledge and Data Engineering* 35 (2022), 12181–12193. https://api.semanticscholar.org/CorpusID:245650768

[13] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining.* 413–422. https://doi.org/10.1109/ICDM.2008.17

[14] R.T. Ng and Jiawei Han. 2002. CLARANS: a method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering* 14, 5 (2002), 1003–1016. https://doi.org/10.1109/TKDE.2002.1033770

[15] Ronald K Pearson. 2006. The problem of disguised missing data. *ACM SIGKDD Explorations Newsletter* 8, 1 (2006), 83–92.

[16] Python Package Index 2023. *pycountry 23.12.11.* Python Package Index. https://pypi.org/project/pycountry/.

[17] Abdulhakim A. Qahtan, Bader Alharbi, Sarah Al-Mutairi, et al. 2018. FAHES: A Robust Disguised Missing Values Detector. In *Proceedings of the KDD '18 Conference.* https://dl.acm.org/doi/10.1145/3219819.3220109

[18] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient Algorithms for Mining Outliers from Large Data Sets. *ACM SIGMOD Record* 29, 427–438.

[19] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. BIRCH: an efficient data clustering method for very large databases. *SIGMOD Rec.* 25, 2 (jun 1996), 103–114. https://doi.org/10.1145/235968.233324

[20] Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research* 20, 96 (2019), 1–7. http://jmlr.org/papers/v20/19-011.html