American Journal of Interdisciplinary Research and Development
ISSN Online: 2771-8948
Website: www.ajird.journalspark.org
Volume 38, March - 2025

THE IMPORTANCE OF DATA CLEANING METHODS FOR 3D VISUALIZATION OF NETWORK DATABASES

Norqulova Ziyoda Nabi qizi
Assistant Teacher of TSEU,
ziyodanorqulova@mail.ru

Abstract
The complex structure and large size of network databases make it difficult to effectively visualize them. This article highlights the importance of data cleaning methods in the process of depicting network data in 3D format. The article considers the stages of identifying incorrect, duplicate and unnecessary data, their correction and standardization. Using cleaned and accurate data, it is shown how to improve the quality of the visualization process and effectively analyze it through a graphical model of the network. This article presents new approaches to 3D visualization and data management from a scientific and practical perspective.

Keywords: Data cleaning, 3D visualization, network database, data quality, big data processing, data preprocessing, data imputation.

Introduction

Modern networks and large-scale databases (Big Data) are widely used in various fields. However, their complexity and uncertainty make data analysis and visualization processes difficult. Data Cleaning is an important step to ensure the reliability of data and facilitate the process of their analysis. Especially when visualizing databases in the network in 3D format, the accuracy of data is crucial for creating visual models.

This article examines the technical aspects of the Data Cleaning process for network data, methods for correcting incorrect and missing data, as well as the advantages of data prepared for 3D visualization. It also examines the theoretical significance and practical approaches of this process, and analyzes the impact of the results on visualization. This approach allows for effective analysis and decision-making in the field of 3D visualization.

Four Stages of Data Cleaning:

Screen	Lack of data Excess of data Outliers Inconsistencies Strange patterns Suspect analysis results
Diagnose	Data is missing Errors Valid records:True extremes No diagnosis:still suspect
Treat	Leave unchanged Correct Delete
Document	Maintain change log Archive raw data and old values

Data Cleaning methods:

1. Various methods are used for data cleaning. The most important ones are listed below:

Identifying and Filling in Missing Data:

- Imputation of missing values using statistical methods.
- Imputation by referencing the data source.

2. **Correcting incorrect values:**

- Identify and filter outliers.
- Find and correct logical errors.

3. **Remove duplicate entries:**

- Identifying duplicates through identifiers.
- Consolidate return data.

4. **Formatting and standardization:**

- Ensuring that data is in a uniform format.
- Making dates, currencies, or units of measurement look the same.

For example:
If the value is missing because it doesn't exist at all, for example:

- The customer has never taken a loan, so their credit score is unavailable;
- The student has not been admitted yet, so their grades are not available;
- The company has not opened a physical store, so its offline sales volume is unavailable;

- The employee has just been hired, so there is no information about their previous projects;
- The product has not been manufactured yet, so its market price is unavailable;

If the values are completely unavailable, attempting to estimate them would not be logically correct. These values should be stored as NaN. On the other hand, if the values are missing due to a lack of entry, it may be possible to infer them based on other values in the same column and row.

If missing rows need to be dropped, this can be done using the *dropna()* function in Pandas:

nfl_data.dropna()

Similarly, the *fillna()* function can be used to replace NaN values with zero.

Another highly effective method is **Scaling** – a preprocessing technique used to transform data into a specific range (e.g., 0-1 or 0-100). The goal of scaling is to ensure that all features contribute equally to the model's performance, especially for distance-based algorithms like k-nearest neighbors (KNN) and support vector machines (SVM).

Without scaling, features with larger numerical ranges can dominate distance calculations, leading to incorrect results. For example, consider product prices expressed in both Japanese Yen and US Dollars. Since 1 US Dollar is approximately 100 Yen, failing to scale the prices would cause methods like SVM or KNN to treat a 1 Yen price difference as significant as a 1 US Dollar difference, which contradicts our intuitive understanding of the real world.

There are different types of scaling:

Scaling is a preprocessing technique used to transform data so that it fits within a specific range (e.g., 0-1 or 0-100). The goal of scaling is to ensure that all features contribute equally to a model's performance, especially for distance-based algorithms like k-nearest neighbors (KNN) and support vector machines (SVM). Without scaling, features with larger numerical ranges may dominate the distance calculations, leading to biased results. For example, you might be looking at the prices of some products in both Yen and US Dollars. One US Dollar is worth about 100 Yen, but if you don't scale your prices, methods like SVM or KNN will consider a difference in price of 1 Yen as important as a difference of 1 US Dollar! This clearly doesn't fit with our intuitions of the world. Types of scaling are there:

1. Min-Max Scaling:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

2. Standard Scaling (Z-Score Normalization):

$$z = \frac{x - \mu}{\sigma}$$

American Journal of Interdisciplinary Research and Development
ISSN Online: 2771-8948
Website: www.ajird.journalspark.org
Volume 38, March - 2025

3. Robust Scaling:

$$x' = \frac{x - \text{median}(x)}{\text{IQR}}$$

Additionally, **normalization** is very useful for optimizing data structure, reducing data redundancy, and minimizing anomalies. It involves dividing data into multiple related tables and linking them through logical relationships. A well-normalized data structure is easier to understand and use efficiently, making analysis more convenient.

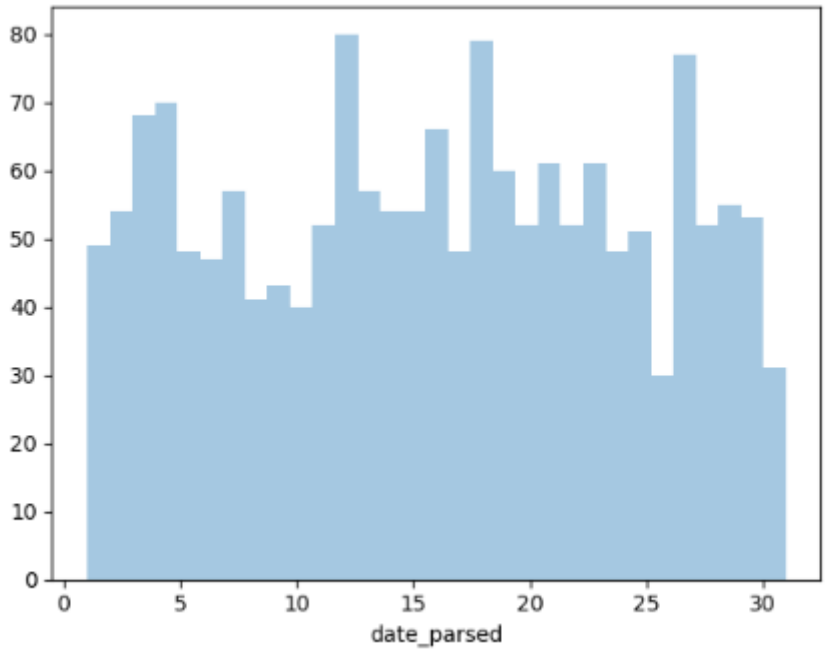
It is also important to convert date columns into the **datetime** format. To do this, the first step is to determine the format of the dates. The key idea is to identify which parts of the date correspond to specific components and the delimiters used between them. While there are many possible date components, the most common ones include %d for the day, %m for the month, %y for a two-digit year, and %Y for a four-digit year.

Some examples:

- 1/18/24 has the format "%m/%d/%y"
- 18-1-2024 has the format "%d-%m-%Y"

we can see that it's in the format "**month/day/two-digit year**".

One of the biggest pitfalls in date analysis is confusing months and days. To prevent this error, a histogram of months and days can be created. We anticipate the values will range from 1 to 31, and, as there's no reason to believe landslides occur more frequently on certain days of the month than others, the distribution should be fairly uniform. (With a slight decrease at 31 since not every month includes 31 days):



Another important step is character encoding. Character encoding is the process of converting characters (letters, numbers, symbols, etc.) into a format that can be stored and processed by computers. In databases, it ensures that textual data is represented

consistently and correctly across different systems. Character encoding is so important because consistency, multilingual support, data integrity and interoperability. Character encoding maps characters to unique binary representations (bytes). For example, ASCII or UTF-8 encoding. if we try to convert a string to bytes for ASCII using *encode (),* we can ask for the bytes to be what they would be if the text was in ASCII.

Additional significant aspect is the Inconsistent Data Entry process. Inconsistent data entry refers to situations where data is entered into a system in multiple formats or styles, making it difficult to analyze, process, or maintain. It often occurs when there are no strict guidelines or validations for data input, resulting in variations that lead to ambiguity or errors. Common Causes of Inconsistent Data Entry:

1. Lack of Standardization: No predefined formats for data input (e.g., dates, names, or addresses).
2. Human Error: Typing mistakes, spelling variations, or failure to follow conventions.
3. Multiple Sources of Data: Data from different systems or contributors using varying formats.
4. Case Sensitivity Issues: Variations in capitalization (e.g., "John Doe" vs. "john doe").
5. Different Measurement Units: Mixing units (e.g., "5kg" vs. "11 pounds").
6. Ambiguous Formatting: Different ways to represent the same data (e.g., "10/12/2024" vs. "December 10, 2024").

We can Use fuzzy matching to correct inconsistent data entry.

Problems Caused by Inconsistent Data Entry

1. Reduced Data Quality: Inconsistencies make the data less reliable for decision-making.
2. Difficulty in Searching or Sorting: Variations make it harder to locate or categorize data.
3. Data Duplication: Inconsistent entries can create duplicates (e.g., "Jane Doe" and "J. Doe").
4. Errors in Analysis: Inconsistent formats can lead to incorrect results in calculations or visualizations.
5. Increased Processing Time: Cleaning and standardizing data consumes additional resources.

How to Prevent or Address Inconsistent Data Entry Standardization:

1. Establish clear data entry guidelines for all users (e.g., date format: YYYY-MM-DD).
2. Validation Rules:Implement rules to check for acceptable formats during data entry (e.g., dropdowns for country names).
3. Training:Educate users about the importance of consistent data entry.
4. Data Cleaning Tools:Use software tools to identify and resolve inconsistencies (e.g., removing duplicates or correcting formats).

5. Automation:Automate data collection with structured forms or APIs to minimize manual entry.

Additionally, we're going to use the fuzzywuzzy package to help identify which strings are closest to each other. This dataset is small enough that we could probably could correct errors by hand, but that approach doesn't scale well. (Would you want to correct a thousand errors by hand? What about ten thousand? Automating things as early as possible is generally a good idea. Plus, it's fun!) Fuzzy matching: The process of automatically finding text strings that are very similar to the target string. In general, a string is considered "closer" to another one the fewer characters you'd need to change if you were transforming one string into another. So "apple" and "snapple" are two changes away from each other (add "s" and "n") while "in" and "on" and one change away (replace "i" with "o"). You won't always be able to rely on fuzzy matching 100%, but it will usually end up saving you at least a little time.

References

1. Rahm E., Do H.H.Data cleaning:problems and current approaches. IEEE Data engineering Bulletin,23(4),3-13.2000.
2. Han J., Pei J., Kamber M. Data mining:Concepts and techniques. Elsevier.2011
3. Hamroqul o'g'li, Arabov Ubaydullo, and Eshonqulov Hamza Ilhomovich. "Katta ma'lumotlar (big data) ni tahlil qilish usullari." PEDAGOGIK MAHORAT 5.1 (2021): 197-201.
4. Bekmirzayeva M. S. Videokuzatuv tizimlari ma'lumotlarini sun'iy intellekt yordamida tahlil qilish: Videokuzatuv tizimlari ma'lumotlarini sun'iy intellekt yordamida tahlil qilish //MODERN PROBLEMS AND PROSPECTS OF APPLIED MATHEMATICS. – 2024. – Т. 1. – №. 01.
5. Beknazarova, S. S., and M. Sh Bekmirzayeva. "Study of the Degree of Influence of Fog on Images in Visualization Systems." (2024).
6. Norqulova, Ziyoda. "DATA VISUALIZATION STEPS TO SORT DATA BY REMOVING MISSING VALUES AND CONVERTING DATA TO NUMERIC TYPE." DTAI–2024 1.DTAI (2024): 37-39.
7. Norqulova, Ziyoda. "MA'LUMOTLARNI VIZUALIZATSIYA QILISH UCHUN ASOSIY TEXNOLOGIYALAR HAMDA ULARNING ONLINE PLATFORMALARDA QO'LLANILISHI. VIZUAL DATA MINING." DIGITAL TRANSFORMATION AND ARTIFICIAL INTELLIGENCE 1.2 (2023): 179-184.