

A Brief Introduction to Data Preprocessing

Huiyi Wu^a

Warwick Mathematics Institute, University of Warwick, Coventry, CV4 7AL, U.K.

Keywords: Data Preprocessing, Big Data, Data Cleaning, Machine Learning.

Abstract: The contemporary digital era is characterized by an exponential growth in data volume. However, this proliferation is paradoxically met with a decline in raw data quality, with most datasets suffering from inconsistencies, missing values, and noise. This paper argues that data preprocessing is the indispensable discipline that bridges the gap between low-quality raw data, and reliable, high-performance machine learning models. Specifically, this essay will explore the multifaceted process of data preprocessing, examining its four primary stages: data cleaning, which removes inaccuracies; data integration, which harmonizes disparate data sources; data transformation, which normalizes and structures data; and data reduction, which increases storage and computational efficiency. Special emphasis will be placed on the unique challenges of preparing unstructured, non-numerical data for analysis, detailing the conversion techniques required to make them compatible with quantitative models. Ultimately, this essay demonstrates that a thorough understanding of data preprocessing is the foundational bedrock upon which all effective, data-driven insights are built.


1 INTRODUCTION

Machine learning traces its roots back to the dawn of artificial intelligence in the 1950s. Simultaneously, the field of data science was evolving from traditional statistics, focusing on learning from data instead of classical mathematical statistics. Exploratory Data Analysis (EDA), Data Preprocessing (DP), and Feature Engineering (FE) are distinct but related stages in the data science workflow. The primary difference lies in their objective and outcome. EDA is the initial investigation phase, corresponding to "Data Understanding." Its purpose is to analyze and summarize a dataset to uncover patterns, spot anomalies, and form hypotheses, typically through statistical summaries and visualizations, without actually changing the source data. Following this, DP and FE fall under "Data Transformation," where the data is actively modified. Data Preprocessing emphasises processing and cleaning the raw data so that it may be used with a model, while Feature Engineering is the intellectual process of using the data that already exists to create new, more insightful features to boost a model's predictive power.

The history of data preprocessing evolved from manual clerical work to a sophisticated, automated

science. Early forms consisted of manual "data editing", where statisticians and census takers would visually inspect tables for errors. A foundational shift towards automation occurred with the work of Fellegi and Holt (Fellegi & Holt, 1976), who, in 1976, introduced a mathematical theory for automatic editing and imputation, providing the first systematic approach for computer-based data cleaning. This formalisation was adopted and expanded in the 1990s with the rise of data warehousing, where the "Transform" stage of the Extract, Transform, Load (ETL) process became a dedicated preprocessing step to ensure data quality for business intelligence (Kimball & Ross, 2013). Today, in the era of machine learning, preprocessing is a critical discipline encompassing techniques from feature scaling to advanced imputation, recognised as essential for model performance and codified in modern data science libraries.

In essence, data preprocessing is the foundational cleanup that ensures data quality and reliability, creating a solid base for both feature engineering and the final modeling phase.

^a <https://orcid.org/0009-0007-0530-4643>

2 DATA PREPROCESSING

In classification with neural networks, this process can eliminate irrelevant data to produce faster learning due to a smaller data set and reduced confusion caused by those data. Overall, get more presentable data through data preprocessing. There is some overlap between data preprocessing and feature engineering, for example, dealing with duplication and missing values. However, data preprocessing deals with raw and imperfect data so that the data set is understandable and useful, and feature engineering is considered an iterative process that creates optimal features. The key techniques in data preprocessing include Data integration, Data cleansing, Data normalization, Data reduction, Data transformation.

2.1 Data Integration

Data integration refers to the process of combining data from multiple sources into a single, unified database. This process involves identifying and accessing different data sources, converting the data to a common format, and reconciling any inconsistencies between sources. As a result, when users integrate data, they can successfully keep clear of issues that limit the precision and speed of machine learning performance.

The traditional method to achieve this data integration is called ETL (Extract, Transform, Load). EFL is a framework that collects a series of tools, including uniform resource identifier (URI), resource description framework (RDF), protege ontology editor and so on (Bansal, 2014). EFL describes the process of extracting data from external sources, transforming data to fit operational requirements and load into the target database. Take a step into the three processes: extraction involves selecting data from appropriate sources, where data is usually available in flat file formats like csv, xls; transformation activities cover data normalization, data filtering, sorting and grouping data which will carry on talking about later in this review; then the process of loading is the propagation of data into a data mart or data warehouse where stores high-quality and accessible data. A data warehouse is a multidimensional view of databases that is easy for data summary and data analysis. Here are some books to read more about data warehousing, discussing the integration of data warehousing to support modern data-driven decision-making (Taniar & Rahayu, 2022).

Application Programming interfaces (APIs) integration is a process of connecting multiple applications or systems to exchange data and perform

actions (Web & Dominte, 2023). There are many types of them and are widely used in different areas from mobile apps to the operating system on the laptop or even a fridge. These characteristics lead to a result of having plenty of types of APIs, such as push/stream APIs, native APIs, SDKs APIs and REST (representational state transfer). REST API is one of the alternative solutions for data exchange and therefore data integration. There are three main design precepts, addressability, uniform interface and statelessness. The goal of REST API is to transmit and receive data using JSON format (Javascript Object Notation) (Alimuiddin et al., 2020).

2.2 Data Cleansing

Broadly, most real-life data sets are dirty and imperfect, including missing data, wrong data and non-standard representation of the same data. Those dirty data negatively impact the performance, for example, bias, inaccurate, or unreliable predictions. In depth, it can cause overfitting or underfitting which is harder to spot. Because of this, filthy data is typically displayed as either noisy data or missing values (MV).

Missing values can usually be handled in three different ways:

Discard the examples with MVs. There are two ways to achieve this, listwise deletion and pairwise deletion which means you can remove the entire rows containing any missing values or remove MVs for specific analysis while keeping the rest of the row data for other analysis. However, this method can reduce the size of the data set so it is only considered when there is a small number of MVs. The more popular way to solve the problem is the second method below.

Imputation of MVs. One of the main advantages of this technique is that it doesn't depend on the learning algorithm being utilised. Therefore, users can choose the most appropriate methods to deal with MVs regardless of what model is chosen for ML. There are numerous imputation techniques available, ranging from fundamental ones like k-nearest neighbours (KNN) and mean substitution, to more advanced approaches that explore inter-attribute relationships. These include methods based on support vector machines (SVM), clustering algorithms, logistic regression, maximum likelihood estimation, and multiple imputation strategies. These approaches are commonly used for quantitative data since they are easily adaptable to other knowledge areas (García, Luengo, & Herrera, 2015).

Noise in data describes random error in raw data. There are two primary categories of noise: attribute noise and class noise. The former speaks of data that is incorrectly labeled which can be introduced by contradictory labeling or misclassifications, whereas the latter case is when distortions are found in values of attributes, including insufficient, missing, or unidentified attribute values. The performance of the models is closely tied to the quality of datasets and the resilience of the model itself to noise. Hence, one of the main tasks in data cleaning is identifying noises in a data set. Numerous strategies have been investigated to handle noisy data, and the more important ones are noise filters, data polishing methods and robust learners. Robust learners are designed to be less affected by noisy data by using pruning techniques to prevent overfitting. However, even strong learners may perform poorly if the noise level is excessive. The goal of data cleaning techniques is to eliminate noisy examples prior to training, but are most effective for small datasets due to their time-consuming nature. According to research, performance can be enhanced by correcting noise in training data while leaving test data noisy. Noise filters, on the other hand, detect and remove noisy instances from the training data, and are particularly helpful for those who are noise-sensitive.

2.3 Data Normalization

Data normalization is part of a preprocessing approach where the data is either scaled or transformed to make an equal contribution to each feature. This is to improve data quality and furthermore improve the performance of the machine learning algorithm. The data normalization step transforms features into a common range so that greater numeric feature values cannot dominate small features (Singh & Singh, 2020). However, be aware that this process does not imply equal importance of the features. Some features are tightly correlated to others, whereas others are superfluous and completely irrelevant. This is a problem to be solved in data reduction.

As mentioned, there are two main tasks in data normalization, re-scaling dominant features and outliers. Mean and Standard Deviation Normalization Methods and Min-Max Value Based Normalization are two methods that are broadly used since they are simple and effective in most cases, and the former method performs particularly better when outliers are present. However, there are also decimal scaling normalization, median and median absolute deviation normalization and tanh based normalization.

In Mean and Standard Deviation Normalization, to normalise the data, the statistical mean and standard deviation are applied. For example, z-score normalization is where the raw data is rescaled such that zero mean and unit variance characterise the resulting features. These techniques aid in minimising the impact of data outliers. Moreover, for Min-Max value based normalization, typically, the data is rescaled to fall between 0 and 1 or -1 and 1. These methods preserve the relationships among the original input data, unlike mean and standard deviation methods that could change over time. However, the biggest issue with these methods is their vulnerability to extreme values and outliers.

2.4 Data Reduction

Data reduction is the process of minimizing the size of datasets while preserving the essential information by reducing the redundant and irrelevant data, or by summarizing the data into more concise form. The definition sounds very similar to data cleaning. However, data cleaning focuses more on fixing errors, inconsistencies and inaccuracies, while data reduction aims to decrease the size and complexity of data. In a short word, data cleaning improves data quality, whereas data reduction simplifies analysis and storage. Dimensionality reduction, numerosity reduction, and cardinality reduction are the three primary categories into which basic data reduction techniques can be divided. Examples of each technique are PCA and t-SNE, sampling and clustering, and encoding and discretization respectively.

Dimensionality reduction focuses on reducing the number of features or random variables in the data set. The main algorithms established into feature selection and feature extraction. Feature selection identifies and removes irrelevant and redundant information to obtain a subset of features. This algorithm reduces the risk of overfitting (García et al., 2016). Feature extraction reduces the number of dimensions by generating a whole new set of features by combining the original ones. One of the baseline approaches is Principal Components Analysis (PCA). Sample numerosity reduction extracts an alternative smaller data representation for the original data. There are either parametric or non-parametric methods. Cardinality reduction applies transformations to obtain a reduced representation of the original data. Discretization is one of the broadly used techniques in ML. This process transforms quantitative data into qualitative data, i.e. numerical attributes into discrete attributes. The main job of

discretization, or data reduction, is necessary since the majority of machine learning algorithms now in use are made to learn categorical data, such as nominal attributes, real-world apps typically incorporate continuous features and this transition is done within data reduction. There is a significant overlap between data reduction and data transformation since the idea of constructing a new set of components is similar.

2.5 Data Transformation

Data transformation is a critical process in data management and analytics, involving the conversion of data from one format or structure into another to enhance its usability and compatibility. Therefore, it is not a surprise that data transformation is integrated with another process have mentioned earlier. This process is essential for ensuring data quality, consistency, and readiness for analysis, particularly in environments where raw data is often unstructured or inconsistent. According to Kimball and Ross, data transformation is a fundamental step in data warehousing, where extracted data undergoes cleaning, normalization, and aggregation before being loaded into a target system (Kimball & Ross, 2013). Modern businesses rely on data transformation to integrate disparate data sources, enabling comprehensive insights through business intelligence (BI) tools and machine learning models. Common transformation techniques include data cleansing (removing duplicates or errors), standardization (converting data into a uniform format), enrichment (augmenting data with additional information), and aggregation (summarizing data for reporting). With the rise of big data, technologies such as Apache Spark, SQL-based ETL (Extract, Transform, Load) tools, and cloud-based data pipelines have become instrumental in automating and scaling transformation processes. Effective data transformation not only improves analytical accuracy but also supports regulatory compliance by ensuring data adheres to organizational and legal standards. As organizations increasingly adopt data-driven decision-making, the role of data transformation continues to grow, making it a cornerstone of successful data strategies (Inmon, 2005).

3 NON-NUMERICAL DATA PREPROCESSING

3.1 Image

Numerical data is typically structured in formats such as CSV and Excel, where each row indicates an observation and each column a feature. However, Image data are normally defined in two-dimensional. Unlike structured numerical data, images require specialized techniques due to their high dimensionality, spatial relationships, and pixel-level variations. The first step typically involves resizing and cropping to standardize dimensions, ensuring uniformity across the dataset, which is crucial for neural networks that require fixed input sizes. Color space conversion is another common preprocessing step; for instance, converting RGB images to grayscale reduces the computational load when color information is unnecessary, while transformations to HSV or LAB spaces can improve feature extraction in certain tasks. Normalization is essential to scale pixel values (typically 0-255) to a range of $[0,1]$ or $[-1,1]$ to facilitate faster convergence during training. Noise reduction techniques, such as Gaussian blurring or median filtering, help mitigate artifacts caused by sensor imperfections or compression, enhancing image clarity.

Data augmentation artificially expands the training dataset by applying transformations like rotation, flipping, zooming, and brightness adjustments. This technique helps prevent overfitting and improves model generalization, especially in scenarios with limited labeled data. For deep learning applications, contrast enhancement methods like histogram equalization or adaptive thresholding can improve feature visibility, particularly in medical imaging or low-light conditions. Additionally, edge detection, e.g., using Sobel or Canny filters may be employed to highlight structural features before feeding images into traditional machine learning models. In cases where images contain irrelevant backgrounds, segmentation or masking techniques isolate regions of interest, reducing noise and focusing the model on relevant features.

The particular task determines which preprocessing procedures are used, for instance, facial recognition systems may prioritize alignment and illumination correction, while autonomous vehicles rely on robust noise reduction and edge detection. Proper preprocessing not only optimizes model accuracy but also reduces training time and

resource consumption, making it indispensable in modern computer vision workflows.

3.2 Text

Text preprocessing is required in natural language processing (NLP) that transforms raw textual data into a structured format suitable for machine learning models. Unlike numerical or image data, text requires specialized techniques to handle linguistic variability, noise, and high dimensionality. The first step typically involves noise removal, where irrelevant characters (e.g., HTML tags, punctuation, or special symbols) are eliminated to retain only meaningful content. Tokenisation then allows for more detailed analysis by breaking the text down into smaller components like words or subwords; this can be language-specific, as in the case of compound words in German or morphologically rich languages like Arabic. Lowercasing is often applied to ensure uniformity, though it may discard meaningful capitalization (e.g., "Apple" as a company vs. the fruit). For languages with inflectional morphology, lemmatization (reducing words to their base forms using dictionaries) or stemming helps normalize variations like "running" → "run". Stop word removal filters out high-frequency but low-information words (e.g., "the," "and") to reduce noise, though this may harm tasks like sentiment analysis where context matters. Vectorization converts text into numerical representations, with traditional methods like bag-of-words (BoW) or TF-IDF (term frequency-inverse document frequency) capturing word importance. Modern approaches leverage word embeddings (e.g., Word2Vec, GloVe) or contextual embeddings (e.g., BERT, ELMo) to encode semantic relationships. For low-resource languages, transliteration or machine translation may bridge preprocessing gaps. Dimensionality reduction (e.g., PCA, LSA) can further refine feature spaces, especially for high-dimensional BoW models.

4 FEATURE ENGINEERING AND RESTRICTIONS IN THIS AREA

Preprocessing, while essential for handling noise and inconsistencies, introduces irreversible transformations—normalizing or scaling data, for instance, may obscure interpretability, particularly in linear models where coefficients rely on original units. Techniques like imputation for missing values or outlier removal can inadvertently introduce bias, as

assumptions (e.g., mean imputation) may not reflect true data distributions. For text and image data, preprocessing choices (e.g., stop-word removal or resizing) may discard semantically or structurally meaningful information, limiting model adaptability. Computational constraints also pose restrictions; complex FE pipelines or high-dimensional transformations (e.g., kernel methods) may be infeasible for large-scale datasets without significant resources. Ultimately, practitioners must balance creativity in FE with caution in preprocessing to avoid compromising model robustness, fairness, or deployability.

5 CONCLUSIONS

The practice of data preprocessing is as old as data collection, although its formalisation as a distinct field is a more recent development. Its history is not one of a single invention but an evolution reflexed by the ever-growing scale and complexity of data, from hand-checked paper ledgers to the automated pipelines of machine learning. In conclusion, data preprocessing is a fundamentally critical and non-negotiable stage in any machine learning pipeline. Its core purpose remains unchangeable across all data types: to clean, transform, and structure raw and chaotic data into a high-quality, efficient format that machine learning algorithms can interpret. For images, preprocessing moves beyond simple cleaning but spatial transformation like resizing and cropping. For text, preprocessing focuses on deconstructing language into numerical representation, like tokenization, identifying stop words and vectorizing words. Preprocessing is the crucial preparatory work that directly impacts models' accuracy, efficiency and reliability. Data preprocessing is the bedrock upon which effective data-driven insights are built.

REFERENCES

- Alimuddin, A. A., Niswar, M., Indrabayu, & Rijal, M. (2020). Design and implementation of REST API for academic information system. IOP Conference Series: Materials Science and Engineering, 875(1), Article 012047.
- Bansal, S. K. (2014, June). Towards a semantic extract-transform-load (ETL) framework for big data integration. In 2014 IEEE International Congress on Big Data (pp. 522–529). IEEE.
- Fellegi, I. P., & Holt, D. (1976). A systematic approach to automatic edit and imputation. *Journal of the American Statistical Association*, 71(353), 17–35.

- García, S., Luengo, J., & Herrera, F. (2015). Data preprocessing in data mining (Vol. 72, pp. 59–139). Cham, Switzerland: Springer International Publishing.
- García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, 1, 1–22.
- Han, J., Kamber, M., & Pei, J. (2012). *Data mining: Concepts and techniques*. Waltham: Morgan Kaufmann Publishers.
- Inmon, W. H. (2005). *Building the data warehouse*. John Wiley & Sons, Incorporated.
- Kimball, R., & Ross, M. (2013). *The data warehouse toolkit: The definitive guide to dimensional modeling*. John Wiley & Sons.
- Singh, D., & Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, 97, 105524.
- Taniar, D., & Rahayu, W. (2022). *Data warehousing and analytics: Fueling the data engine*. Poland: Springer International Publishing.
- Web, A. P. I., & Dominte, I. (2023). *Web API development for the absolute beginner*.

