International Journal of Core Engineering & Management

Volume-6, Issue-12, 2021 ISSN No: 2348-9510

LEVERAGING MACHINE LEARNING AND DATA VAULT MODELING FOR AUTOMATED MASTER DATA MANAGEMENT

Gautham Ram Rajendiran
gautham.rajendiran@icloud.com

Abstract

Master Data Management (MDM) plays a critical role in ensuring data consistency and accuracy across an organization's information systems. Traditional MDM approaches often rely on rule-based matching or manual curation, which can be time-consuming and error-prone. This paper introduces a novel methodology that leverages the Data Vault 2.0 modeling approach in combination with Machine Learning (ML) algorithms to automate MDM processes and address data quality issues such as duplication, inconsistency, and incorrect mappings. By integrating hubs and same-as links, the proposed solution identifies and groups similar records, allowing real-time updates to MDM and enhancing the integrity and scalability of the data platform.

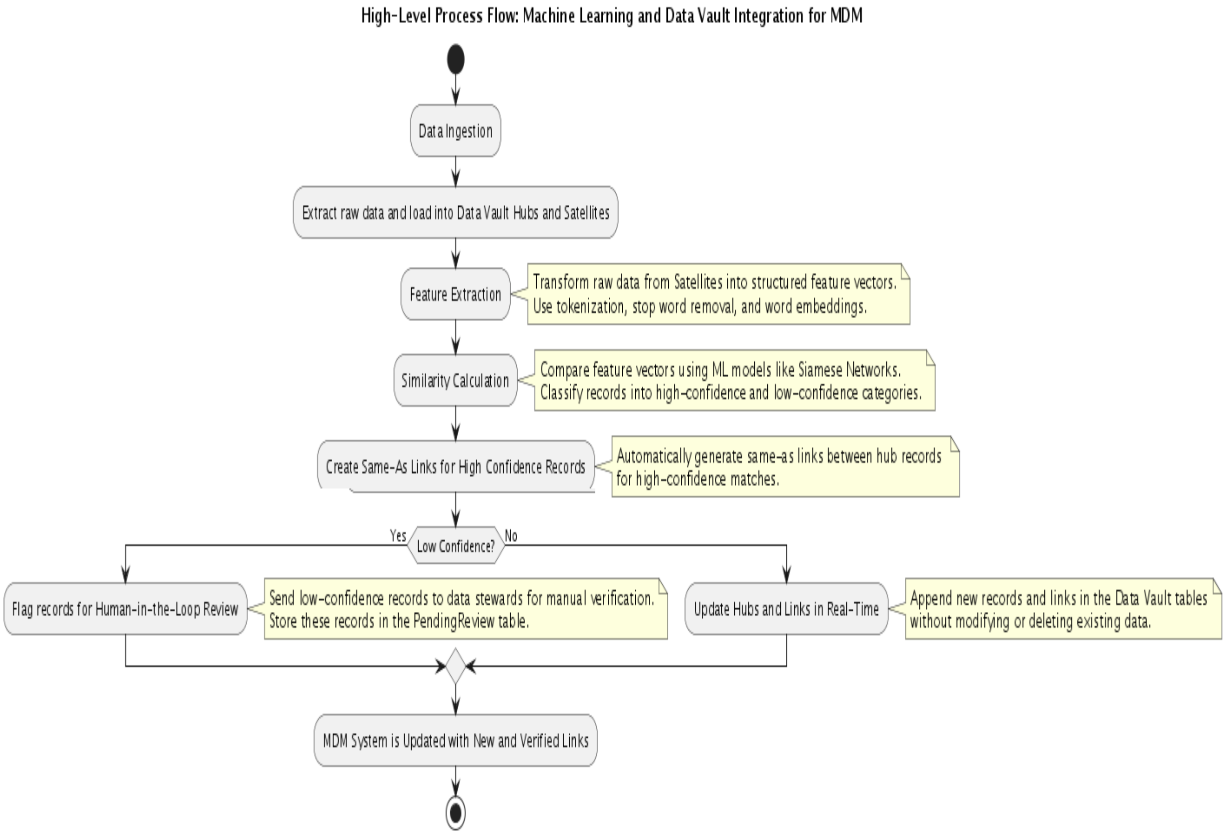
Keywords: Machine Learning, Master Data Management, Data Vault, Data Management, Data Engineering

I. INTRODUCTION

Organizations generate enormous volumes of data daily from multiple sources, including customer interactions, product inventories, supplier information, and transaction logs. Ensuring data quality and consistency across these disparate sources is a significant challenge. Master Data Management (MDM) [1] is designed to address this challenge by providing a unified view of core business entities like customers, products, and suppliers. However, existing MDM systems often struggle to keep pace with growing data volumes and complexity [2], especially when changes to master data occur frequently.

The Data Vault 2.0 modeling methodology, with its emphasis on scalability and agility, offers a robust architecture to capture, store, and maintain master data in a manner that suits the dynamic nature of modern enterprises. The append-only nature of Data Vault makes it a perfect candidate for event-driven architectures, enabling real-time updates to master data as soon as new information is received. This paper presents a detailed methodology for implementing an enhanced MDM system using Data Vault and machine learning, providing step-by-step insights into building a scalable and automated MDM solution.

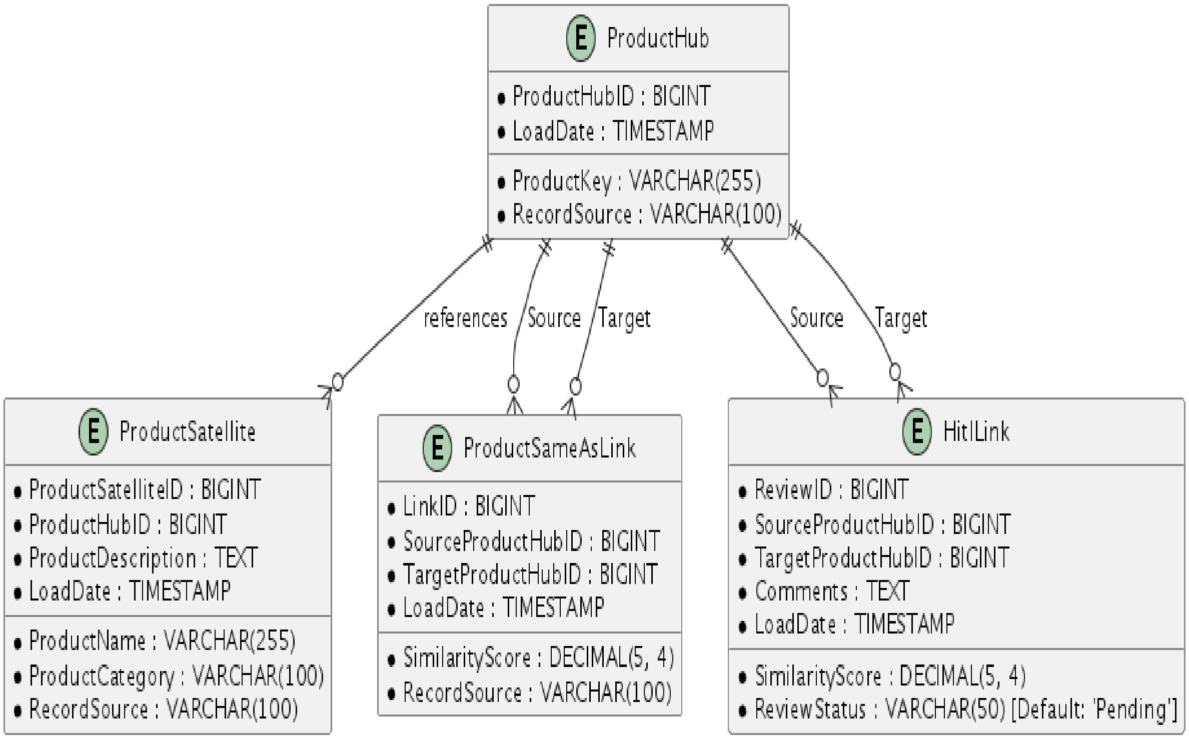
II. METHODOLOGY


High-Level Process Flow: Machine Learning and Data Vault Integration for MDM
Data Ingestion
Extract raw data and load into Data Vault Hubs and Satellites
Feature Extraction
Transform raw data from Satellites into structured feature vectors.
Use tokenization, stop word removal, and word embeddings.
Similarity Calculation
Compare feature vectors using ML models like Siamese Networks.
Classify records into high-confidence and low-confidence categories.
Create Same-As Links for High Confidence Records
Automatically generate same-as links between hub records
for high-confidence matches.
Yes
Low Confidence?
No
Flag records for Human-in-the-Loop Review
Send low-confidence records to data stewards for manual verification.
Store these records in the PendingReview table.
Update Hubs and Links in Real-Time
Append new records and links in the Data Vault tables
without modifying or deleting existing data.
MDM System is Updated with New and Verified Links

The proposed methodology for enhancing MDM with Data Vault and Machine Learning involves several key steps, starting with defining the Data Vault structures, extracting features for machine learning, and creating same-as links for record grouping. The steps and implementation details are elaborated in the following sections by using ecommerce product identifiers as a sample use-case.

2.1 Data Vault Schema

Shown in Fig2 is a data vault schema that can be used to create a master data management mechanism for various product identifiers such as UPC, ASIN and others.


E ProductHub
ProductHubID : BIGINT
LoadDate : TIMESTAMP
ProductKey : VARCHAR(255)
RecordSource : VARCHAR(100)
references
Source
Target
Source
Target
E ProductSatellite
ProductSatelliteID : BIGINT
ProductHubID : BIGINT
ProductDescription : TEXT
LoadDate : TIMESTAMP
ProductName : VARCHAR(255)
ProductCategory : VARCHAR(100)
RecordSource : VARCHAR(100)
E ProductSameAsLink
LinkID : BIGINT
SourceProductHubID : BIGINT
TargetProductHubID : BIGINT
LoadDate : TIMESTAMP
SimilarityScore : DECIMAL(5, 4)
RecordSource : VARCHAR(100)
E HitlLink
ReviewID : BIGINT
SourceProductHubID : BIGINT
TargetProductHubID : BIGINT
Comments : TEXT
LoadDate : TIMESTAMP
SimilarityScore : DECIMAL(5, 4)
ReviewStatus : VARCHAR(50) [Default: 'Pending']

The core structure of a Data Vault model for MDM consists of Hubs, Links, and Satellites.

Hubs store the unique business keys that identify entities such as customers, products, or suppliers. Each Hub captures only the business key and metadata such as creation timestamps and record sources. The `ProductKey` in this table could be an SKU, ASIN, or any other identifier used within the organization's systems. This key will serve as the foundation for linking additional information through same-as links and Satellites.

Same-as links are used to define relationships between different hub records that may represent the same real-world entity but have variations due to data quality issues or inconsistent formats. For example, a same-as link can indicate that ProductHubID1 and ProductHubID2 represent the same product entity with a certain confidence level. This structure captures the relationship between two hub records, the similarity score, and metadata like record source and load date. For example, if ProductHubID1 is identified as being similar to ProductHubID2 with

85% confidence, the ProductSameAsLink table will store a row indicating this similarity relationship.

Satellites store descriptive attributes related to the hub keys, such as product name, category, and description. These attributes are later used to compute similarity scores between records. This table allows for historical tracking of descriptive attributes while preserving the integrity of the core business key in the Hub.

2.2 Feature Extraction for Machine Learning

Feature extraction is a critical process that transforms raw data stored in the Satellite tables into structured inputs for machine learning models. Each hub's descriptive attributes, stored in Satellite tables, are used to create feature vectors that capture the characteristics of each entity. The extracted features are then used to calculate similarities between records.

For example, consider a product dataset where each product has a \`ProductName\`, \`ProductCategory\`, and \`ProductDescription\`. Textual attribute vectorization [3] is a process used to convert text data, such as product names and descriptions, into numerical representations that can be processed by machine learning models. This transformation is crucial for similarity detection in MDM systems, as it allows models to understand the underlying patterns and relationships between different records. The implementation of this process involves multiple steps, including tokenization [4], removing stop words [5], generating word embeddings [6], and constructing feature vectors [7]. Each of these steps is detailed below, along with an explanation of their impact on the data model and entity-relationship diagram (ERD).

Tokenization is the initial step in textual vectorization, where the raw text data is broken down into individual tokens, which are typically words or phrases. For instance, consider a product description such as "Wireless Bluetooth Headphones." The tokenization process will split this description into separate components: ["Wireless", "Bluetooth", "Headphones"]. Each token is treated as an independent entity, capturing a distinct piece of the product's description.

The relationship between ProductToken and ProductHub is established through the ProductHubID, ensuring that each token is linked to its corresponding product entity. To reflect tokenization in the ERD, a new entity named ProductToken is introduced. The ProductToken entity captures individual tokens extracted from the ProductSatellite entity, along with metadata such as token frequency, position, and context. This structure allows the system to store and manage tokens separately, facilitating more granular analysis and similarity calculations.

Next step is to remove stop words, Stop word removal is the process of eliminating common words (e.g., "the", "is", "at") that do not contribute significant meaning to the context of a product description. Removing these words reduces the dimensionality of the feature space, making the vectorization process more efficient and meaningful.

In the ERD, stop word removal can be represented as a filtering step applied to the ProductToken entity, where tokens flagged as stop words are excluded from further processing. This filtering ensures that only relevant tokens are retained in the ProductToken entity for each product.

Word embeddings are dense vector representations of words that capture semantic relationships and contextual similarities between different tokens. Techniques such as Word2Vec [8] or BERT [9] are used to convert each token into a multi-dimensional vector. For example, the word "wireless" might be represented as a 300-dimensional vector: [0.12, -0.08, 0.33, ...]. These embeddings allow the system to understand that "Bluetooth" and "wireless" are contextually similar, even if they do not share common letters.

The token vectors are used to construct a Feature Vector [7]. Feature vector construction involves aggregating the token vectors for each product entity to create a comprehensive feature vector that represents the entire product. This aggregation can be done using techniques such as averaging, summation, or concatenation of the individual token vectors. The resulting feature vector serves as the input for the machine learning model to perform similarity detection.

The ProductFeatureVector entity links back to the ProductHub entity through the ProductHubID, ensuring that each feature vector is associated with its corresponding product. This structure allows for efficient similarity calculations and ensures that each product’s characteristics are captured comprehensively.

The constructed feature vectors are used as inputs to machine learning models for similarity detection. During similarity calculation, the system compares feature vectors between different products to identify relationships such as high-confidence equal, low-confidence equal, high-confidence not equal, and low-confidence not equal. Depending on the confidence levels, appropriate same-as links are created, or the records are flagged for review in the PendingReview table.

The updated entity relation diagram including the newly defined entities are shown in Fig3

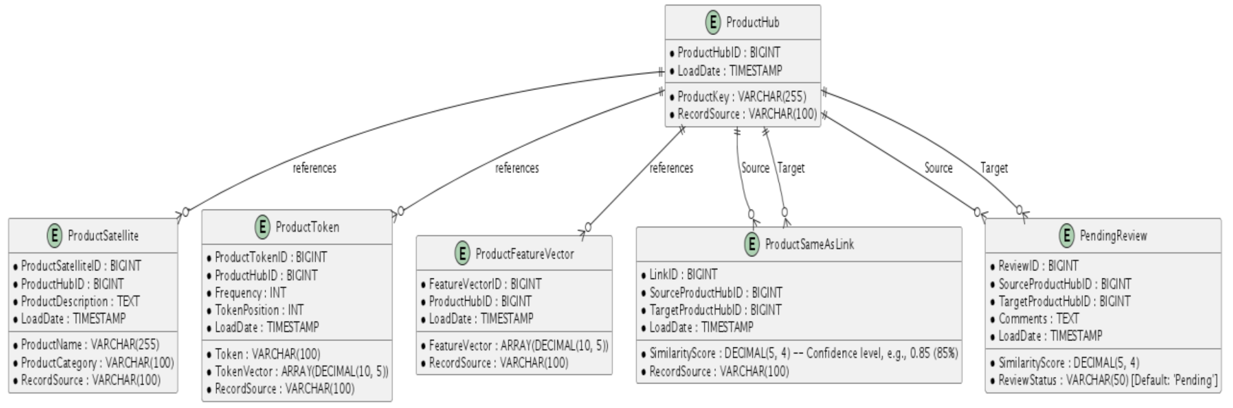

E ProductHub
ProductHubID : BIGINT
LoadDate : TIMESTAMP
ProductKey : VARCHAR(255)
RecordSource : VARCHAR(100)
references
references
references
Source
Target
Source
Target
E ProductSatellite
ProductSatelliteID : BIGINT
ProductHubID : BIGINT
ProductDescription : TEXT
LoadDate : TIMESTAMP
ProductName : VARCHAR(255)
ProductCategory : VARCHAR(100)
RecordSource : VARCHAR(100)
E ProductToken
ProductTokenID : BIGINT
ProductHubID : BIGINT
Frequency : INT
TokenPosition : INT
LoadDate : TIMESTAMP
Token : VARCHAR(100)
TokenVector : ARRAY(DECIMAL(10, 5))
RecordSource : VARCHAR(100)
E ProductFeatureVector
FeatureVectorID : BIGINT
ProductHubID : BIGINT
LoadDate : TIMESTAMP
FeatureVector : ARRAY(DECIMAL(10, 5))
RecordSource : VARCHAR(100)
E ProductSameAsLink
LinkID : BIGINT
SourceProductHubID : BIGINT
TargetProductHubID : BIGINT
LoadDate : TIMESTAMP
SimilarityScore : DECIMAL(5, 4) -- Confidence level, e.g., 0.85 (85%)
RecordSource : VARCHAR(100)
E PendingReview
ReviewID : BIGINT
SourceProductHubID : BIGINT
TargetProductHubID : BIGINT
Comments : TEXT
LoadDate : TIMESTAMP
SimilarityScore : DECIMAL(5, 4)
ReviewStatus : VARCHAR(50) [Default: 'Pending']

Fig 3 Updated Entity Relationship Diagram With Tables for Machine Learning Metadata

By incorporating textual attribute vectorization and embedding these processes in the ERD, the proposed MDM framework can handle complex product descriptions, identify similarities more effectively, and improve the overall quality of master data management. This approach enhances the system's capability to automatically detect and resolve data quality issues, such as duplicate or inconsistent records, while providing a foundation for advanced analytics and machine learning applications.

III. RESULTS

The proposed framework for enhancing Master Data Management (MDM) using Data Vault modeling and Machine Learning algorithms introduces a novel approach to automating entity matching and similarity detection. The framework utilizes a robust feature extraction process, similarity classification, and dynamic same-as link creation to address common data quality issues such as duplication and inconsistency. By incorporating machine learning models to categorize relationships into four confidence levels—high-confidence equal, low-confidence equal, high-confidence not equal, and low-confidence not equal—the system intelligently automates the grouping of records while providing a mechanism for human review through a human-in-the-loop [10] process.

The ability of the framework to classify records based on confidence levels allows for a nuanced approach to similarity detection. High-confidence matches can be automatically linked using same-as links, while low-confidence matches are flagged for manual review, thereby ensuring that the MDM system maintains a high standard of accuracy and quality without overwhelming data stewards with unnecessary manual curation. The event-driven nature of the Data Vault architecture, combined with its append-only structure, further enables real-time updates to the MDM system, making it a promising solution for dynamic and fast-changing environments.

The results discussed are based on theoretical modeling and anticipated outcomes derived from existing research and practices in MDM and machine learning. Future work and experimentation are required to validate the effectiveness and scalability of the proposed solution in real-world scenarios.

IV. CONCLUSION

This paper presents a detailed conceptual framework for enhancing Master Data Management by integrating Data Vault modeling with advanced Machine Learning techniques. By leveraging hubs, satellites, and same-as links in the Data Vault architecture, the proposed methodology automates the detection and grouping of similar records, thereby improving the efficiency and quality of MDM processes. The use of machine learning models to classify records into high-confidence and low-confidence categories enables a hybrid approach where automation and human verification work in tandem to ensure the accuracy of the master data repository.

The framework's alignment with event-driven architectures allows for real-time updates to the MDM system, providing a significant advantage over traditional MDM approaches that often rely on batch processing. While the results discussed are theoretical, the potential benefits of implementing this solution are clear, including reduced manual data curation, improved data quality, and enhanced scalability.

Future research should focus on implementing this framework in a real-world context to evaluate its performance and scalability. Additionally, exploring the use of more advanced machine learning models, such as deep learning and graph neural networks, could further refine the accuracy and capabilities of the MDM system. By advancing this research, organizations can develop more intelligent and automated solutions for managing master data, ultimately leading to better data-driven decision-making and operational efficiency.

REFERENCES

1. Loshin, David. Master data management. Morgan Kaufmann, 2010.
2. Pansara, Ronak. "Master Data Management Challenges." International Journal of Computer Science and Mobile Computing 10.10 (2021): 47-49.
3. Neshatian, Kourosh, and Mahmoud R. Hejazi. "Text categorization and classification in terms of multi-attribute concepts for enriching existing ontologies." 2nd Workshop on Information Technology and its Disciplines. 2004.
4. Grefenstette, Gregory. "Tokenization." Syntactic wordclass tagging. Dordrecht: Springer Netherlands, 1999. 117-133.
5. Silva, Catarina, and Bernardete Ribeiro. "The importance of stop word removal on recall values in text categorization." Proceedings of the International Joint Conference on Neural Networks, 2003.. Vol. 3. IEEE, 2003.
6. Gutiérrez, Luis, and Brian Keith. "A systematic literature review on word embeddings." Trends and Applications in Software Engineering: Proceedings of the 7th International

Conference on Software Process Improvement (CIMPS 2018) 7. Springer International Publishing, 2019.

7. Bellet, Aurélien, Amaury Habrard, and Marc Sebban. "A survey on metric learning for feature vectors and structured data." arXiv preprint arXiv:1306.6709 (2013).
8. Church, Kenneth Ward. "Word2Vec." Natural Language Engineering 23.1 (2017): 155-162.
9. Kenton, Jacob Devlin Ming-Wei Chang, and Lee Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." Proceedings of naacL-HLT. Vol. 1. 2019.
10. Zanzotto, Fabio Massimo. "Human-in-the-loop artificial intelligence." Journal of Artificial Intelligence Research 64 (2019): 243-252.