U.PORTO

MASTER
MODELING, DATA ANALYTICS AND DECISION SUPPORT SYSTEMS

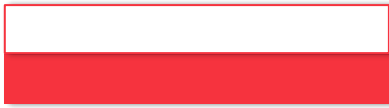
RECORD LINKAGE PROBLEM

Bruno Miguel Couto da Silva

M

2024

FACULDADE DE **ECONOMIA**



RECORD LINKAGE PROBLEM

Bruno Miguel Couto da Silva

Dissertation
Master in Modeling, Data Analytics and Decision Support Systems

Supervised by
Professor João Manuel Portela Gama

2024

Abstract

This thesis was created within the scope of the Master in Modeling, Data Analysis and Decision Support Systems, at the Faculty of Economic, University of Porto. It was based on a Report Linkage Problem proposed by BPLim. Record Linkage is the process of linking records in a database to the same real-life entity. The aim of this project was to review the available bibliography on Record Linkage and case studies, to develop an algorithm that would solve the problem at hand.

In the Record Linkage Problem at hand, a case of connecting several databases from different supermarkets to match products through their descriptions will be presented. Eleven different methods for comparing these records were found in the bibliography reviewed. From these eleven methods found, four were selected based on their reliability, efficiency and relevance to the case. These four methods were combined to build a more comprehensive and efficient algorithm that would solve the Record Linkage Problem.

With the objective of developing the best method to solve the problem introduced, the sensibility and parameterization of the algorithm were evaluated using the preestablished and well validated metrics: precision, recall, f-score. The algorithm developed fulfilled the criteria defined to solve the Record Linkage Problem.

Keywords

Record Linkage, strings, Lenvshtein, Q-grams, Cosine, R, Smith-Waterman, Jaro-Winkler, Pre-processment, Text Mining, Data Mining.

Index

Figure index

Table index

Equation index

Glossary

Record Linkage: The process of identifying and merging records from different sources that refer to the same entity.

Matching Algorithm: A computational procedure used to determine if two records refer to the same entity.

Blocking: A technique to reduce the number of record comparisons by partitioning records into blocks based on common attributes.

Clerical Review: The manual inspection of record pairs to verify matches, often used to validate automated record linkage results.

False Positive: A record pair incorrectly identified as a match.

False Negative: A record pair incorrectly identified as a non-match.

Precision: The proportion of true matches among all the record pairs identified as matches.

Recall: The proportion of true matches that were correctly identified out of all true matches.

F-Score: A harmonic mean of precision and recall, providing a single metric to evaluate linkage quality.

Probabilistic Record Linkage: A method that calculates the probability that records match based on observed similarities and differences.

Deterministic Record Linkage: A method that uses exact or rule-based criteria to link records.

Jaro-Winkler Distance: A string comparison metric that accounts for typographical errors and transpositions.

Levenshtein Distance: A metric for measuring the difference between two sequences by counting the minimum number of operations required to transform one sequence into the other.

Feature Engineering: The process of creating new features from raw data to improve the performance of machine learning models.

Standardization: The process of transforming data into a common format to facilitate comparison and integration.

Data Cleaning: The process of detecting and correcting (or removing) corrupt or inaccurate records from a dataset.

Similarity Measure: A function that quantifies the similarity between two records, often used in the context of record linkage.

Training Data: A dataset used to train machine learning models, containing examples of both matched and non-matched record pairs.

Receiver Operating Characteristic (ROC) Curve: A graph showing the performance of a classification model at all classification thresholds.

Area Under the Curve (AUC): A single scalar value summarizing the performance of a model, derived from the ROC curve.

Scalability: The capability of a system to handle a growing amount of work or its potential to accommodate growth.

Big Data: Extremely large datasets that may be analyzed computationally to reveal patterns, trends, and associations.

Data Integration: The process of combining data from different sources to provide a unified view.

1 Introduction

Over time, data activities have gained relevance in every sector from industry to healthcare. In this thesis, a more specific approach will be taken to a data mining process. This approach allowed the interconnection of data from different sources, through measurements applied to the information to interpret its coincidence. This process is called record linkage.

As a first step in the elaboration of this thesis, a bibliography review was conducted., The focus of this review was exploring the concept of record linkage, its process, the difference between deterministic linkage and probabilistic linkage and the evaluation methods used in different contexts of character analysis. This review established the basis for the detailed characterization of the methods that could be used in the case resolution process showing how these are calculated and what each considers to measure similarities.

After exploring the available methodologies and the theoretical basis of Record Linkage, the problem of this case study will be introduced. The various datasets relevant for the case study will be explained, along with the main objective of the desired resolution for this case study.

To determine the optimal approach for resolving the presented issue, the algorithm's sensibility and parameterization were assessed utilizing pre-established and well verified metrics: precision, recall, f-score.

1.1 Motivation and objectives

My motivation for choosing this problem was due to the fact that Record Linkage is a data mining problem Although it has not been extensively explored in this master's classes, it is a very relevant topic in practice. Record linkage is a pivotal process which has great societal impact, it underpins many of the advancements in healthcare, policy development, academic research, and business management. By enabling a more comprehensive and integrated view of data, it drives better outcomes, more efficient operations, and deeper insights. As our society becomes increasingly data-driven, the importance of effective and ethical record linkage will only continue to grow, shaping a future where data can be harnessed to its fullest potential for the benefit of all.

The objectives for this investigation are summarized in the following points:

1. Background and theoretical study of record linkage;
2. Evaluation of similarity measures and record linkage methods;
3. Apply the different methods to real world datasets;
4. Select the best methods.

1.2 Thesis structure

This dissertation provides a comprehensive analysis of record linkage theory and case studies. It reviews related work, detailing the evolution of record linkage from early manual methods to modern automated systems, and introduces key concepts such as blocking, matching, and data integration. The dissertation delves into the mathematical and statistical theories behind record linkage, comparing probabilistic and deterministic methods, and exploring various statistical models and preprocessing steps for data cleaning and standardization. It examines matching techniques, record linkage methods, including character-based and machine learning approaches, and emphasizes the importance of performance metrics. Through case studies, it illustrates the application of record linkage in diverse domains such as healthcare, socioeconomic research, education, criminal justice, and genealogy, while also addressing common issues like duplicate records, missing data, and variations in data formats. Privacy concerns, ethical implications, computational challenges, and solutions are discussed, alongside emerging techniques like deep learning and blockchain technology, and the integration with big data technologies such as Hadoop and Spark. The impact of policy changes and ethical guidelines is also considered. The problem resolution chapter focuses on datasets, preprocessing, proposed solutions, and the evaluation of experimentation, culminating in an analysis of results and conclusions.

2 Related work (Bibliography review)

2.1 What is record linkage

Databases play a very important role in today's economy, with several industries and systems dependent on their accuracy in order to allow better conclusions from the analyzes carried out, allowing to make the best decisions in the operations in question. That said, it is possible to conclude that the quality of databases has a significant impact on a system that depends on information to work and conduct a business (Boyd & Crawford, 2012; Chetty, Friedman, & Rockoff, 2014; Christen P. , 2012; Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008).

The record linkage process consists of identifying pairs of records from different data sources that refer to the same information. The situations that require this approach are usually databases that contain descriptive information about each entity, such as name, address, postal code, serial numbers and others (Christen P. , 2012) (Elmagarmid, Ipeirotis, & Verykios, 2007) (Fellegi & Sunter, 1969) (Talburt, 2011).

However, it is common for there to be variations in data values between different databases that refer to the same entity. Such as, for example, name variations due to spelling errors, an indication of the nickname in a database and the real name in another, situations in which the maiden name was indicated in a dataset and in another the married name was provided and among others. (Christen P. , 2012) (Elmagarmid, Ipeirotis, & Verykios, 2007) (Fellegi & Sunter, 1969) (Talburt, 2011).

Figure 1 Record Linkage concept

The record linkage process boils down to a process of obtaining data from different databases, in which it first goes through the data cleaning and standardization process in

which its main task focuses on converting the raw input data into well defined, consistent forms, as well as the resolution of inconsistencies in the way information is represented and encoded. It should be noted that data cleaning and standardization of text fields such as names, addresses and descriptions is very important in order to avoid introducing redundant information. That said, it goes through an indexing process, moving to a record pair comparison and then applying similarity analysis methods in which the output can be summarized in three results: match, non-match or possible match. In the case of a possible match, a clerical review will be necessary in order to understand whether it is in fact an information match or not. (Christen P. , 2012) (Elmagarmid, Ipeirotis, & Verykios, 2007) (Fellegi & Sunter, 1969) (Talburt, 2011).

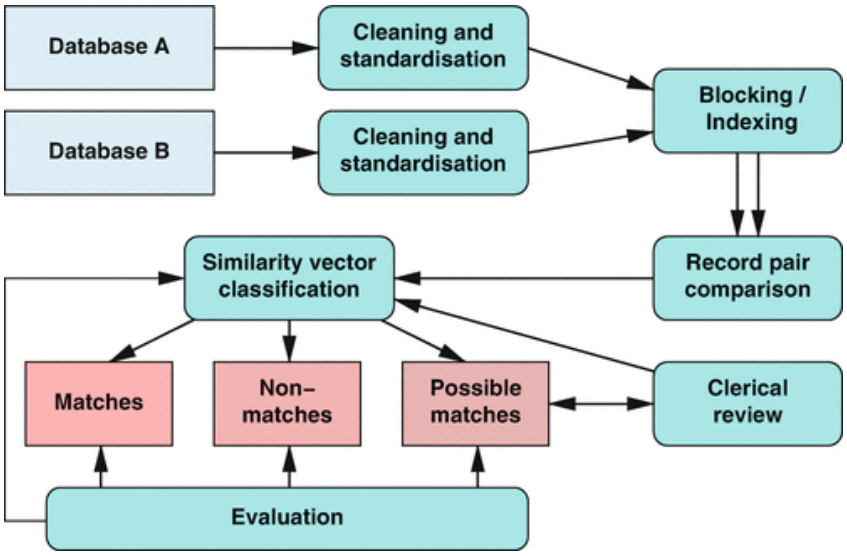


Figure 2 Record Linkage process

The record linkage can have several approaches in which deterministic linkage and probabilistic linkage stand out. Deterministic linkage is the exact matching of information while probabilistic linkage is a method for combining information from records on different datasets to form a new linked dataset. It has been described as a process that attempts to link records on different files that have the greatest probability of belonging to the same person/organization. (Christen P. , 2012) (Fellegi & Sunter, 1969) (Talburt, 2011; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999)

2.2 Evolution of record linkage

2.2.1 Early Manual Methods

Record linkage has evolved significantly since its inception. The early methods were manual and labor-intensive, involving human operators who would compare records from different sources. This process was not only time-consuming but also prone to errors, especially as the volume of data increased. Initially, record linkage was used in census data to track population changes and in healthcare to combine patient records from different hospitals (Dunn, 1946).

2.2.2 Semi-Automated Methods

With the advent of computers in the mid-20th century, semi-automated methods began to emerge. These methods still required significant human intervention but used computer algorithms to perform initial matches, which were then verified by human operators. This reduced the workload and improved accuracy compared to purely manual methods. Techniques such as sorting and matching based on unique identifiers like Social Security numbers became common (Newcombe, Kennedy, Axford, & James, 1959; Winkler W. E., Advanced Methods for Record Linkage, 1994).

2.2.3 Probabilistic Methods

The 1960s and 1970s saw the development of probabilistic methods, most notably the Fellegi-Sunter model. This model introduced a statistical approach to record linkage, assigning probabilities to the likelihood that pairs of records from different datasets referred to the same entity. Probabilistic methods allowed for more sophisticated matching by considering various attributes and their potential errors. This was a significant leap forward, allowing for more accurate and scalable record linkage (Fellegi & Sunter, 1969; Winkler & Thibaudeau, 1991).

2.2.4 Deterministic Methods

Parallel to the development of probabilistic methods, deterministic methods were also refined. These methods relied on exact matches between specific fields (e.g., name, date of birth) to link records. While less flexible than probabilistic methods, deterministic approaches were simpler and faster, making them suitable for situations where high-quality, standardized data were available (Christen P. , 2012).

2.2.5 Modern Automated Systems

The 21st century has seen the rise of fully automated record linkage systems. These systems leverage advanced machine learning algorithms and big data technologies to link records across vast datasets with minimal human intervention. Techniques such as clustering, neural networks, and natural language processing are now used to improve the accuracy and efficiency of record linkage. Additionally, data integration tools and platforms have emerged, offering comprehensive solutions for linking and analyzing data from multiple sources (Fellegi & Sunter, 1969; Winkler & Thibaudeau, 1991).

2.3 Key Concepts in Record Linkage

2.3.1 Blocking

Blocking is a technique used to reduce the number of comparisons needed during the record linkage process. Instead of comparing every record with every other record, which is computationally expensive, records are divided into smaller blocks based on certain attributes (e.g., the first letter of the last name). Comparisons are then only made within these blocks, significantly speeding up the process. Effective blocking strategies are crucial for handling large datasets (Christen P. , 2012; Fellegi & Sunter, 1969).

2.3.2 Matching

Matching refers to the process of determining whether records from different datasets refer to the same entity. This involves comparing attributes such as names, addresses, and dates of birth. Matching can be performed using deterministic methods (requiring exact matches) or probabilistic methods (allowing for partial matches and errors). Modern systems often use a combination of techniques, including machine learning algorithms, to improve matching accuracy (Christen P. , 2012).

2.3.3 Data Integration

Data integration is the process of combining data from different sources to provide a unified view. In the context of record linkage, data integration involves merging matched records into a single dataset, resolving inconsistencies, and standardizing formats. Effective

data integration is essential for ensuring that the linked data is accurate, complete, and usable for analysis (Christen P. , 2012).

2.3.4 Significant Research Papers and Algorithms

The Fellegi-Sunter Model

One of the foundational works in the field of record linkage is the Fellegi-Sunter model, developed by Ivan Fellegi and Alan Sunter in the 1960s. Their model introduced a probabilistic approach to record linkage, focusing on the use of statistical methods to determine the likelihood that two records refer to the same entity (Fellegi & Sunter, 1969; Hernandez & Stolfo, 1995; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999).

Central to their model are key components such as a decision rule for classifying record pairs into matches, non-matches, or potential matches based on calculated probabilities, a system for assigning weights to attributes according to their discriminative power, and techniques for managing errors and data variations (Fellegi & Sunter, 1969; Hernandez & Stolfo, 1995; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999).

The model offers improved accuracy compared to deterministic methods and provides flexibility in handling partial matches and errors. However, its implementation is computationally intensive and necessitates meticulous parameter tuning and threshold setting to achieve optimal results (Fellegi & Sunter, 1969; Hernandez & Stolfo, 1995; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999).

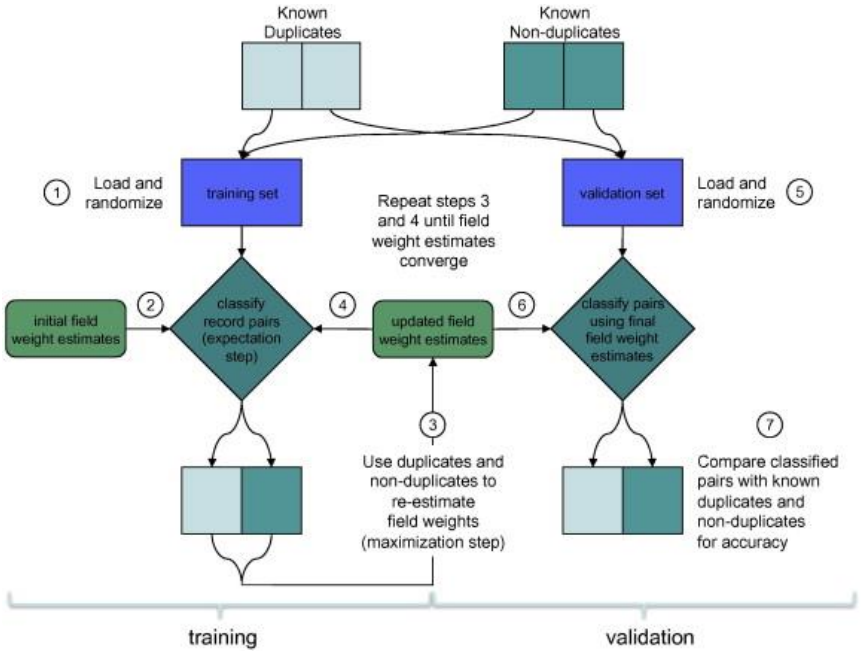


Figure 3 Fellegi-Sunter Model

Sorted Neighborhood Method

Developed by M.A. Hernandez and S.J. Stolfo in the 1990s, the Sorted Neighborhood Method (SNM) involves sorting records based on a key attribute and then only comparing records within a fixed "window" of the sorted list. This approach significantly reduces the number of comparisons required, thereby expediting the matching process.

Its advantages include efficiency with large datasets and straightforward implementation. However, SNM's effectiveness depends heavily on the selection of the sorting key and may overlook matches that fall outside the predefined window, which can affect its overall accuracy in certain contexts (Hernandez & Stolfo, 1995).

Expectation-Maximization (EM) Algorithm

The EM algorithm, used in various probabilistic record linkage methods, iteratively estimates the parameters of a statistical model by maximizing the likelihood of observed data. It is particularly useful in situations where data contains missing or uncertain information (Winkler W. E., Methods for Record Linkage and Bayesian Networks , 2002).

Its benefits include robustness to missing data and the potential to enhance linkage accuracy through iterative refinement. However, implementing the EM algorithm can be computationally demanding. Moreover, it may converge to local optima, necessitating careful consideration of initial parameter estimates to achieve optimal results (Winkler W. E., Methods for Record Linkage and Bayesian Networks , 2002).

2.4 Mathematical and Statistical Theories Behind Record

2.4.1 Linkage

Record linkage involves identifying records that refer to the same entity across different datasets. The mathematical and statistical theories underpinning record linkage focus on measuring the similarity between records and making decisions based on these similarities. Key components include (Christen P. , 2012; Fellegi & Sunter, 1969):

Similarity Measures**:**

- **Edit Distance (Levenshtein Distance):** Measures the minimum number of edits (insertions, deletions, substitutions) needed to transform one string into another.
- **Jaccard Index:** Measures the similarity between two sets by comparing the size of their intersection with the size of their union.
- **Cosine Similarity:** Measures the cosine of the angle between two non-zero vectors, often used for text data represented as term frequency vectors.

Decision Rules:

- **Threshold-based Decision Making:** If the similarity score exceeds a predefined threshold, the records are considered a match.
- **Probabilistic Decision Making:** Uses statistical models to estimate the likelihood that records match, and then decides based on probabilities.

Error Handling:

- **False Matches (Type I Errors):** Incorrectly linking non-matching records.
- **Missed Matches (Type II Errors):** Failing to link matching records.

2.4.2 Probabilistic vs. Deterministic Methods

Probabilistic Methods

Probabilistic record linkage employs statistical models to gauge the likelihood that two records pertain to the same entity, offering flexibility in handling data variations and errors. A prominent example is the Fellegi-Sunter Model, which establishes a mathematical framework where each pair of records is assessed based on comparison vectors for attributes (Fellegi & Sunter, 1969; Christen P. , 2012).

Mathematical Basis:

- Let M be the event that two records match, and U be the event that they do not match.
- Each pair of records has a set of comparison vectors γ, where γi represents the comparison outcome for the *i*-th attribute.
- Calculate match and non-match probabilities: $P(\gamma \mid M)$ and $P(\gamma \mid U)$
- Compute the likelihood ratio: $LR(\gamma) = \frac{P(\gamma|M)}{P(\gamma|U)}$
- Compare $LR(\gamma)$ to two thresholds to classify pairs into matches, non-matches, or potential matches requiring further review.

Its advantages include adeptness with incomplete and noisy data, and enhanced accuracy in the presence of data errors or variations. However, implementing probabilistic record linkage can be computationally demanding and requires meticulous calibration of model parameters to achieve optimal performance (Fellegi & Sunter, 1969; Christen P. , 2012).

Deterministic Methods

Deterministic record linkage relies on exact or rule-based matches on one or more attributes. It is simpler and faster but less flexible. One of those methods is exact matching (Newcombe, Kennedy, Axford, & James, 1959; Winkler W. E., Advanced Methods for Record Linkage, 1994).

Procedure:

- Compare records based on one or more attributes, such as Social Security Number (SSN), name, and date of birth.
- Records are linked if all specified attributes exactly match.

Its advantages include simplicity in implementation and computational efficiency. However, deterministic linkage is inflexible and unable to accommodate variations or errors in data, potentially missing matches due to slight differences in attributes. Therefore, while deterministic methods are quick and effective for exact matches, they may not be suitable for datasets with inherent variability or inconsistencies.

2.5 Statistical Models in Record Linkage

2.5.1 Bayesian Networks

Bayesian networks are graphical models that represent the probabilistic relationships among a set of variables. In record linkage, they can be used to model the dependencies between different attributes and to compute the probabilities of matches (Friedman, Geiger, & Goldszmidt, 1997; Christen P. , 2012).

Application in Record Linkage:

Model Structure:

- Nodes represent attributes (e.g., name, address, date of birth).
- Edges represent dependencies between attributes.

Inference:

- Use Bayesian inference to calculate the posterior probability that two records match, given observed similarities and prior probabilities.

Example:

- Suppose we have attributes AAA (name), BBB (address), and CCC (date of birth).
- The Bayesian network might model the dependency such that AAA and BBB are conditionally independent given CCC.

Computing Probabilities:

- Calculate the joint probability distribution over the attributes.
- Use the observed data to update the probabilities:

$$P(\,match \mid A, B, C\,) = \frac{\big(P(\,A, B, C \mid match\,)P(match)\big)}{P(A, B, C)}$$

Equation 1 Computing probabilities formula

By calculating joint probability distributions over attributes and updating them with observed data, Bayesian networks can effectively handle uncertainties and missing data. However, constructing and validating these networks requires expertise and can be computationally demanding, particularly with large datasets, which may limit their practicality in certain applications (Friedman, Geiger, & Goldszmidt, 1997; Christen P. , 2012).

2.6 Preprocessing Steps for Data Cleaning and Standardization

Preprocessing is a critical step in record linkage to ensure that data is clean, standardized, and ready for accurate matching. The main preprocessing steps include:

Data Cleaning (Rahm & Do, 2000; Christen P. , 2012)**:**

- **Handling Missing Values:** Impute missing data using mean, median, or mode for numerical attributes, and the most frequent value or a placeholder for categorical attributes.
- **Removing Duplicates:** Identify and remove duplicate records within each dataset.
- **Correcting Errors:** Identify and correct errors in data entries, such as typos or formatting inconsistencies.

Data Standardization (Rahm & Do, 2000; Christen P. , 2012)**:**

- **Standardizing Formats:** Ensure uniform formatting for dates, phone numbers, addresses, and other structured data.
- **Normalization:** Convert text data to a consistent case (e.g., all lower case) and standardize abbreviations (e.g., "St." to "Street").

- **Tokenization:** Break down text fields into tokens (words or characters) for more granular comparisons.

Data Transformation (Hernandez & Stolfo, 1995; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999)**:**

- **Encoding Categorical Variables:** Convert categorical variables into numerical codes or dummy variables.
- **Feature Scaling:** Normalize numerical values to a common scale, often using min-max scaling or z-score standardization.

Blocking or Indexing (Hernandez & Stolfo, 1995; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999)**:**

- **Blocking:** Create blocks of records that share common features to reduce the number of comparisons (e.g., using the first letter of the last name).
- **Sorting and Windowing:** Sort records and compare only within a fixed-size window to limit comparisons.

2.7 Matching Techniques

Exact Matching:

- Simple and fast method comparing records based on exact attribute matches (Newcombe, Kennedy, Axford, & James, 1959; Winkler W. E., Advanced Methods for Record Linkage, 1994).

Probabilistic Matching:

- Uses statistical techniques to compute the likelihood that records match, considering the possibility of errors and variations (Fellegi & Sunter, 1969; Christen P. , 2012; Elmagarmid, Ipeirotis, & Verykios, 2007).

Machine Learning-Based Matching:

- Uses supervised or unsupervised learning algorithms to classify pairs of records as matches or non-matches.

2.8 Record linkage methods

The comparison of fields is the key activity of the record linkage process because it is in this phase that the similarity between fields in a unique way is measured. Several algorithms can be applied, and they all have their way of comparing field pairs. There are methods in the context of character analysis, token-based, at the phonetic level and in the numerical context. For the investigation of my thesis, I will only take into account the character-based methods and in the following subchapters, the concepts of each algorithm that will be used and how they are calculated will be explored (Christen P. , A Comparison of Personal Name Matching: Techniques and Practical Issues, 2006; Elmagarmid, Ipeirotis, & Verykios, 2007; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999).

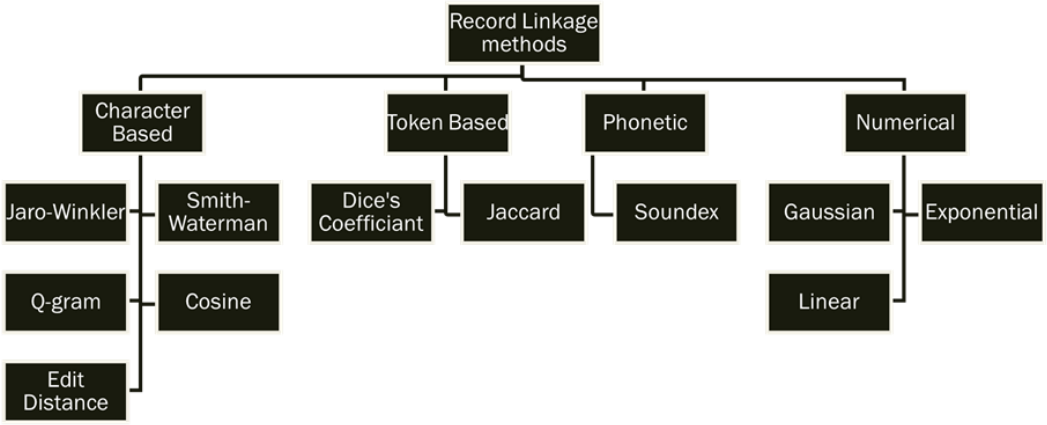


Figure 4. Record linkage methods

2.8.1 Character based methods

The use of character-based method algorithms is applicable in cases of typographical errors, such as scenarios with reference fields such as names, descriptions, addresses and others. This type of algorithm is based on defining how difficult it is to change a string so that it is identical to the other string compared. All algorithms output values between zero and one, where zero represents total dissimilarity and one is an exact match (Christen P. , A Comparison of Personal Name Matching: Techniques and Practical Issues, 2006; Elmagarmid, Ipeirotis, & Verykios, 2007; Navarro, 2001).

Q-gram

The first algorithm is the Q-gram, in which q-grams are short character substrings of length q of the database strings where the similarity measure is calculated by the number

of q-grams in common between two strings in which they are similar when they present a high number of common q-grams (Navarro, 2001; Christen P. , 2012).

Usually, q-grams of q=2 (bigrams) and q=3 (trigrams) are used. When it is assumed that q=1, there is no sensitivity to the order of characters in the strings, for example, if we have a string with the name "Bruno" and another with "Burno" it will be defined which is a perfect similarity (Navarro, 2001; Christen P. , 2012).

The Q-gram's similarity is calculated by the following way:

$$Sim_{qg}(A, B, q) = \frac{|Qgram(A, q) \cap Qgram(B, q)|}{\max(|Qgram(A, q)|, |Qgram(B, q)|)}$$

Equation 2 Q-gram's formula

Cosine

The Cosine similarity is a measure between two non-zero vectors derived from their inner product space. The cosine-angle between the two vectors is computed which gives a measurement of their similarity. The measure takes all of the letters from each name, sorts them alphabetically into unique values, then compares each to the original names to create vectors (Manning, Raghavan, & Schütze, 2008).

The Cosine's similarity is calculated by the following way:

$$Sim_{cs}(A, B) = \frac{A \cdot B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Equation 3 Cosine's formula

Edit Distance

Jaro

The Jaro distance measures the number of common characters between two strings and the number of transpositions. The similarity takes into account the lengths of the two strings (A and B), the number of characters of A and B (m) and the number of transpositions (t) (Winkler & Thibaudeau, 1991; Winkler W. E., Advanced Methods for

Record Linkage, 1994; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999; Winkler W. E., Methods for Record Linkage and Bayesian Networks , 2002).

The Jaro's similarity is calculated by the following way:

$$Sim_{jo}(A,B) = \frac{1}{3}\left(\frac{m}{|A|} + \frac{m}{|B|} + \frac{m - t/2}{m}\right)$$

Equation 4 Jaro's formula

Levenshtein

The Levenshtein distance also measures the number of common characters between two strings, but instead of taking into account the number of transpositions, like Jaro's distance, it allows insertions, deletions and replacements and the output is the smallest number of edit operations required to change one string into another (Navarro, 2001).

The Levenshtein's similarity is calculated by the following way:

$$Sim_{ld}(A,B) = 1.0 - \frac{Levenshtein(A,B)}{\max(|A|,|B|)}$$

Equation 5 Lenvshtein's formula

Jaro-Winkler

Jaro-Winkler distance its a improved version of Jaro's distance made by Winkler, where it gives a higher weight to prefix matches than to the other character matches between the strings (Winkler & Thibaudeau, 1991; Winkler W. E., Advanced Methods for Record Linkage, 1994; Winkler W. E., The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999; Winkler W. E., Methods for Record Linkage and Bayesian Networks , 2002).

The calculation of this similarity takes into account the Jaro's similarity calculated and c values that represents the number of agreeing characters at the beginning of each string and its calculated by the following way (Winkler & Thibaudeau, 1991; Winkler W. E., Advanced Methods for Record Linkage, 1994; Winkler W. E., The State of Record Linkage

and Current Research Problems Statistical Research Division, U.S. Census Bureau., 1999; Winkler W. E., Methods for Record Linkage and Bayesian Networks , 2002):

$$Sim_{jw}(A,B) = Sim_{jo}(A,B) + \frac{c}{10}\Big(1.0 - Sim_{jo}(A,B)\Big)$$

Equation 6 Jaro-Winkler's formula

Smith-Waterman

The Smith-Waterman distance was developed to find optimal aligments between biological sequences. It has a similar approach like levenshtein dsitance where the metric represents the minimum number of changes required to convert one string to another, but it also allows gaps, deletions/insertions of arbitrary length character-specific match scoress (Smith & Waterman, 1981).

The Smith-Waterman's similarity is calculated taking into acccount the highest value within the dynamic programing score matrix (bs_{sw}), the value whent two characters match ($match\_score$), and a factor (div_{sw}) which can be calculated by 3 ways (Smith & Waterman, 1981):

- div_{sw}= min (|A|,|B|);
- div_{sw}= max (|A|,|B|);
- div_{sw}= 0.5 x ((|A|+|B|).

This algorithm has the following formula:

$$Sim_{swd}(A,B) = \frac{bs_{sw}}{div_{sw} \times match\_score}$$

Equation 7 Jaro-Winkler's formula

2.9 Machine Learning Approaches

Machine learning approaches in record linkage have gained significant traction due to their ability to handle large datasets, complex matching rules, and the inherent uncertainty in matching records. Two of these approaches are supervised learning and unsupervised learning (Christen P. , 2012; Mudgal, 2018; Wilson, 2011; Winkler W. E., Advanced Methods for Record Linkage, 1994).

Supervised Learning:

- Requires labeled training data where pairs of records are pre-classified as matches or non-matches.
- Algorithms: Decision Trees, Random Forests, Support Vector Machines (SVM), Neural Networks.

Unsupervised Learning:

- Does not require labeled data; instead, it identifies patterns and clusters in the data.
- Algorithms: K-Means Clustering, Hierarchical Clustering, DBSCAN.

2.10 Importance of Performance Metrics

Performance metrics are essential to evaluate the effectiveness and accuracy of record linkage algorithms. Key metrics include (Christen P. , 2012; Elmagarmid, Ipeirotis, & Verykios, 2007; Winkler W. E., Advanced Methods for Record Linkage, 1994; Fawcett, 2006; Dunn, 1946):

Precision:

- The ratio of true positive matches to the total predicted matches.
- $$Precision = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

Equation 8 Precision formula

Recall:

- The ratio of true positive matches to the total actual matches.
- $$Recall = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

Equation 9 Recall formula

F1 Score:

- The harmonic mean of precision and recall, providing a balance between the two.

- $F1\ Score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Equation 10 F1 Score formula

ROC Curve and AUC:

- The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at various threshold settings.
- The Area Under the ROC Curve (AUC) measures the overall performance of the model.

2.11 Case Studies

2.11.1 Linking Health Records to Improve Patient Care

The study utilized datasets from Electronic Health Records (EHRs) across multiple hospitals, encompassing patient demographics, medical histories, and treatment records, as well as data from national health databases containing standardized patient diagnoses and treatments (Jensen, Jensen, & Brunak, 2012; Westra, 2011).

To link these records, probabilistic record linkage was employed using the Fellegi-Sunter model to manage discrepancies in patient information, along with blocking techniques based on last names and birth years to reduce comparisons. Additionally, machine learning models, specifically Random Forests trained on labeled data, were used to classify record pairs as matches or non-matches (Jensen, Jensen, & Brunak, 2012; Westra, 2011).

The outcomes included successful linking of patient records across hospitals, which improved continuity of care, reduced duplicate records, and ensured comprehensive patient histories were available to healthcare providers. This enhanced the ability to track patient outcomes and treatment effectiveness (Jensen, Jensen, & Brunak, 2012; Westra, 2011).

The impact was significant in healthcare by improving patient care through complete and accurate medical histories, in research by enabling large-scale studies on

treatment outcomes and epidemiology, and in policy by informing health policy decisions with comprehensive patient care data (Jensen, Jensen, & Brunak, 2012; Westra, 2011).

2.11.2 Linking Census Data for Socioeconomic Research

The study utilized datasets from census records spanning multiple decades, which included demographic information, employment data, and housing details, along with tax records providing income and employment details, and social security data containing lifetime earnings and benefits information (Ruggles S. , 2011; Goeken, 2011).

Linkage techniques involved deterministic matching using unique identifiers like Social Security Numbers (SSNs) where available, and probabilistic matching for records without unique identifiers, based on names, birth dates, and addresses. Data standardization ensured consistent formats for names, addresses, and dates across datasets (Ruggles S. , 2011; Goeken, 2011).

The outcomes included successfully linking individuals across multiple decades of census data, creating a comprehensive longitudinal dataset on socioeconomic status, and improving data quality by identifying and rectifying inconsistencies and duplicates (Ruggles S. , 2011; Goeken, 2011).

The impact was significant in economics, providing researchers with rich datasets for studying economic trends and social mobility, in public policy by informing decisions on social welfare, employment, and income inequality, and in sociology by enabling detailed studies on the effects of socioeconomic factors on life outcomes (Ruggles S. , 2011; Goeken, 2011).

2.11.3 Linking Educational Records for Academic Research

The study utilized datasets from school records, including student demographics, grades, and attendance from various school districts, national education databases containing standardized test scores and educational attainment data, and employment records providing post-graduation employment and income information (Chetty, Friedman, & Rockoff, 2014; Reardon, 2017).

Linkage techniques involved deterministic matching using student ID numbers and standardized test scores, and probabilistic matching for cases where student IDs were inconsistent, using names, birth dates, and schools attended. Feature engineering was

applied to create composite features from multiple attributes to improve matching accuracy (Chetty, Friedman, & Rockoff, 2014; Reardon, 2017).

The outcomes included successfully linking student records across different educational stages and employment data, creating a comprehensive dataset for tracking educational attainment and career outcomes over time, and identifying key factors contributing to student success and areas needing intervention (Chetty, Friedman, & Rockoff, 2014; Reardon, 2017).

The impact was significant in education, providing insights into the effectiveness of educational programs and interventions, in policy by informing education policy and funding decisions through identification of successful educational practices, and in research by enabling longitudinal studies on the impact of education on career success and socioeconomic status (Chetty, Friedman, & Rockoff, 2014; Reardon, 2017).

2.11.4 Linking Criminal Justice Records for Legal Studies

The study utilized datasets from police records, including arrest records, incident reports, and demographic information, court records containing case details, verdicts, and sentencing information, and correctional facility records providing incarceration details and release dates (Kling, 2006; Stevens, 2006).

Linkage techniques involved probabilistic matching using names, dates of birth, and social security numbers, as well as blocking based on last name and birth year to reduce the number of comparisons. Machine learning models, employing supervised learning methods, were used to improve matching accuracy and handle variations in data entries (Kling, 2006; Stevens, 2006).

The outcomes included successfully linking criminal justice records across police, court, and correctional facilities to create comprehensive criminal histories, improving data accuracy by identifying and rectifying errors and inconsistencies, and enabling detailed analysis of the criminal justice process from arrest to release (Kling, 2006; Stevens, 2006)..

The impact was significant in legal studies, providing data for studying patterns in criminal behavior and the effectiveness of different justice system interventions, in policy by informing criminal justice reforms through comprehensive data on the justice process,

and in public safety by enabling better tracking of offenders and assessment of rehabilitation programs (Kling, 2006; Stevens, 2006)..

2.11.5 Linking Genealogical Records for Historical Research

The study utilized datasets from historical census records containing demographic information from multiple generations, birth, marriage, and death records providing vital event information, and immigration records including passenger lists and naturalization documents (Ruggles S. , 2006; Larmuseau, 2014).

Linkage techniques involved deterministic matching using unique identifiers like national ID numbers where available, probabilistic matching using names, dates of birth, and places of birth for historical records without unique identifiers, and phonetic matching algorithms such as Soundex to handle variations in name spellings (Ruggles S. , 2006; Larmuseau, 2014).

The outcomes included successfully linking genealogical records across multiple generations and different types of records, creating comprehensive family trees and historical timelines, and improving historical data quality by identifying and correcting inconsistencies (Ruggles S. , 2006; Larmuseau, 2014).

The impact was significant in genealogy by enabling individuals and researchers to trace family histories and understand genealogical connections, in historical research by providing rich datasets for studying historical demographic trends and migration patterns, and in cultural studies by enhancing understanding of cultural and societal changes over time (Ruggles S. , 2006; Larmuseau, 2014).

2.12 Common Issues in Record Linkage

2.12.1 Duplicate Records

Duplicate records occur when the same entity is represented multiple times in a dataset, often with slight variations in details, resulting from data entry errors, system migrations, or the integration of multiple data sources (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

Examples:

- John Doe, J. Doe, and Johnathan Doe might all refer to the same person.
- Multiple patient records due to different hospital visits without a unified patient ID.

Detection and resolution techniques include standardization, which involves uniform formatting for names, dates, and addresses (e.g., converting all text to lowercase, standardizing date formats), and phonetic matching algorithms like Soundex or Metaphone to match names phonetically. Additionally, machine learning models can be trained to identify and merge duplicates based on patterns in the data (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

2.12.2 Missing Data

Missing data occurs when certain values are absent from the dataset, significantly impairing the record linkage process by reducing the accuracy of matches. Examples include absent birth dates in patient records and incomplete address information in customer databases. These gaps hinder the linkage process by limiting the attributes available for comparison, potentially resulting in missed or inaccurate matches. Addressing missing data requires careful consideration of imputation techniques or data augmentation strategies to enhance the completeness and reliability of record linkage outcomes (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

Handling missing data involves several techniques: imputation, which replaces missing values with statistical estimates (e.g., mean, median) or predictions from models; omission, which excludes records with critical missing data, although this can lead to data loss; and data augmentation, which supplements missing data with information from other reliable sources (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

2.12.3 Variations in Data Formats

Different systems often store data in varying formats, leading to inconsistencies that complicate linkage efforts.

For instance, discrepancies in date formats such as MM/DD/YYYY versus DD/MM/YYYY, or variations in address formats based on country or region, complicate the matching process. These inconsistencies can lead to errors or mismatches when

comparing records from disparate sources (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

Standardization techniques to address these issues include data transformation, which converts all data into a uniform format using predefined rules and scripts, and regular expressions, which parse and standardize complex data fields like addresses and phone numbers (Christen P. , 2012; Newcombe, Kennedy, Axford, & James, 1959).

.

2.13 Privacy Concerns and Ethical Implications

2.13.1 Privacy Concerns

Record linkage often involves sensitive data, especially in domains like healthcare, finance, and criminal justice, making the protection of individuals' privacy paramount. Examples include linking health records across hospitals while safeguarding patient identities and merging financial records from diverse institutions securely (Sweeney, 2002; Dwork, 2008; Union, 2016).

Privacy-preserving techniques include de-identification and anonymization, which remove or mask personal identifiers, and encryption, which secures data during transfer and storage to prevent unauthorized access. Additionally, secure multi-party computation allows parties to collaboratively compute functions over their data without revealing their inputs, and differential privacy ensures that the inclusion or exclusion of a single record does not significantly affect the outcome of any analysis, thus protecting individual privacy (Sweeney, 2002; Dwork, 2008; Union, 2016).

2.13.2 Ethical Implications

Ethical concerns surrounding the use of linked data center on the responsible handling of information to respect individual rights and societal norms (Reardon, 2017; Zook, 2017).

Instances include the misuse of linked health records for discriminatory practices and unauthorized linking of data for surveillance or profiling purposes (Reardon, 2017; Zook, 2017).

Ethical guidelines to address these concerns include obtaining informed consent from individuals before using their data, ensuring transparency by clearly communicating the purposes and methods of data linkage, implementing accountability measures to oversee the ethical use of linked data, and upholding fairness to ensure that the benefits of data linkage are equitably distributed and do not disproportionately harm any group. These principles are crucial in promoting ethical standards and safeguarding the privacy and rights of individuals affected by linked data practices (Reardon, 2017; Zook, 2017).

2.14 Computational Challenges and Solutions

2.14.1 Challenges

Dealing with large datasets poses significant computational challenges in linking data, particularly when datasets contain millions of records. The complexity of advanced linkage algorithms, such as probabilistic and machine learning-based methods, further adds to the computational burden. Additionally, in applications requiring real-time processing, like fraud detection or real-time health monitoring, data must be linked quickly and efficiently to derive timely insights (Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008; Zaharia, 2010).

2.14.2 Solutions

To address these challenges, scalable architectures and efficient algorithms are crucial:

Scalable Architectures (Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008; Zaharia, 2010):

- Distributed Computing: Utilize frameworks like Apache Hadoop or Apache Spark to distribute the processing load across multiple nodes, enabling parallel processing and efficient utilization of resources.
- Cloud Computing: Leverage cloud services to benefit from on-demand scalability and computational power, allowing flexibility in handling varying workloads.

Efficient Algorithms (Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008; Zaharia, 2010):

- Blocking and Indexing: Reduce the number of comparisons by dividing data into smaller, manageable blocks based on pre-defined criteria, such as geographical proximity or shared attributes.
- Approximate Matching: Implement algorithms like Locality-Sensitive Hashing (LSH) that balance accuracy and efficiency by focusing on identifying approximate matches rather than exact matches.

Parallel Processing (Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008; Zaharia, 2010):

- GPU Acceleration: Employ GPUs for parallel processing of large datasets, which can significantly accelerate computations due to their ability to handle parallel tasks efficiently.
- Multi-threading: Implement multi-threaded algorithms that effectively utilize multiple CPU cores, enhancing processing speed and efficiency.

Incremental Linking (Dean & Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, 2008; Zaharia, 2010):

- Incremental Algorithms: Continuously update linked datasets as new data arrives, avoiding the need to reprocess the entire dataset from scratch and enabling real-time updates.
- Online Learning: Utilize online machine learning algorithms that can update models incrementally with new data, allowing adaptive learning and continuous improvement of linkage models over time.

These solutions enable efficient and scalable linkage of large datasets, addressing computational challenges while supporting real-time and accurate data processing in various applications.

2.15 Emerging Techniques in Record Linkage

2.15.1 Deep Learning

Deep learning techniques, particularly neural networks, are increasingly being applied to record linkage to improve accuracy and handle complex, high-dimensional data (Manning, Raghavan, & Schütze, 2008; Mudgal, 2018; Wilson, 2011).

Applications include using entity embeddings to transform categorical data (e.g., names, addresses) into dense vectors that capture semantic similarities, thereby facilitating more effective record comparisons downstream (Manning, Raghavan, & Schütze, 2008; Mudgal, 2018; Wilson, 2011)..

Autoencoders are employed to learn compressed representations of data, aiding in identifying similar records by comparing encoded vectors (Manning, Raghavan, & Schütze, 2008; Mudgal, 2018; Wilson, 2011)..

Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs) handle sequential data, capturing dependencies in fields like addresses and names crucial for linking records where order matters (Manning, Raghavan, & Schütze, 2008; Mudgal, 2018; Wilson, 2011)..

These techniques offer scalability for processing large datasets efficiently and can capture intricate patterns and dependencies that traditional methods may overlook. However, they require substantial amounts of labeled training data and demand significant computational resources, posing challenges in implementation (Manning, Raghavan, & Schütze, 2008; Mudgal, 2018; Wilson, 2011)..

2.15.2 Blockchain Technology

Blockchain provides a decentralized and secure method for managing records, ensuring data integrity and provenance, which can be leveraged in record linkage (Nakamura, 2017; Zyskind, Nathan, & Pentland, 2015).

It supports immutable records by storing hashed versions on the blockchain, guaranteeing traceability and preventing unauthorized modifications while linking records using hash comparisons without exposing sensitive information (Nakamura, 2017; Zyskind, Nathan, & Pentland, 2015).

Decentralized data management enables multiple entities to contribute and verify records independently, facilitated by smart contracts that automate linkage processes based on predefined rules (Nakamura, 2017; Zyskind, Nathan, & Pentland, 2015).

The technology enhances data security and transparency by providing a clear history of data changes and linkages (Nakamura, 2017; Zyskind, Nathan, & Pentland, 2015).

However, challenges include scalability issues with current blockchain solutions being slow and costly for large-scale data processing, and the complexity of integrating blockchain with existing data systems and processes, which requires careful consideration and expertise (Nakamura, 2017; Zyskind, Nathan, & Pentland, 2015).

2.16 Integration with Big Data Technologies

2.16.1 Hadoop

Hadoop is an open-source framework that allows for the distributed processing of large datasets across clusters of computers using simple programming models (White, 2015).

In record linkage, Hadoop's applications include leveraging the MapReduce framework to execute blocking and matching algorithms concurrently across a cluster, effectively distributing the workload to handle large datasets efficiently (White, 2015).

The Hadoop Distributed File System (HDFS) supports the storage of vast datasets and intermediate linkage results. Its advantages lie in scalability, managing substantial data volumes by spreading processing tasks across multiple nodes, and cost-effectiveness through the use of standard hardware, reducing infrastructure expenses (White, 2015).

Challenges include the complexity associated with Hadoop and MapReduce programming, requiring specialized expertise, and potential latency issues due to its batch processing nature, which may limit real-time application capabilities (White, 2015).

2.16.2 Spark

Apache Spark is a unified analytics engine for large-scale data processing, known for its speed and ease of use (Zaharia, 2010).

In the realm of record linkage, Spark offers several applications: Resilient Distributed Datasets (RDDs) allow for efficient storage and in-memory processing of extensive datasets, thereby accelerating linkage operations (Zaharia, 2010).

DataFrames and Spark SQL provide structured data processing capabilities, facilitating seamless linkage tasks. Additionally, MLlib, Spark's machine learning library, supports the development and training of models specifically tailored for record linkage (Zaharia, 2010).

Spark's advantages include its speed, enabled by in-memory processing, which significantly reduces computation time, and its flexibility to handle diverse data processing needs and integrate seamlessly with other big data tools (Zaharia, 2010).

However, challenges include the complex management of resources such as memory and computational power, and the learning curve associated with mastering Spark's APIs and data processing paradigms (Zaharia, 2010).

2.17 Policy Changes and Ethical Guidelines

2.17.1 Policy Changes

Data Protection Regulations: Enhancing regulations such as the General Data Protection Regulation (GDPR) to bolster privacy and security measures in record linkage. Global implementation of similar policies aims to standardize data protection practices across jurisdictions (Union, 2016).

Data Sharing Agreements: Developing robust data sharing agreements that clearly define the roles and limitations regarding data usage among entities involved in record linkage. Emphasis is placed on transparency and accountability to safeguard the integrity and ethical handling of linked data (Wilson, 2011).

Standards and Best Practices: Establishing industry-wide standards for data linkage processes to uphold consistency and reliability. Promotion of best practices in data management, cleaning, and linkage aims to enhance overall data quality and adherence to ethical guidelines (Little & Rubin, 2019).

2.17.2 Ethical Guidelines

Informed Consent: Mandating explicit informed consent from individuals before utilizing their data in record linkage projects. This ensures individuals are fully aware of how their data will be used and understand the potential risks involved (Zook, 2017).

Transparency and Accountability: Maintaining transparency throughout the data linkage processes, including how data is collected, linked, and used. Implementing robust mechanisms for auditing and accountability helps prevent misuse and ensures responsible handling of linked data (Westra, 2011).

Equity and Fairness: Upholding principles of equity and fairness by ensuring that the benefits derived from data linkage are distributed fairly and do not disproportionately impact any particular group. Addressing potential biases in data and linkage algorithms is crucial to prevent discrimination and ensure ethical conduct in record linkage practices (Reardon, 2017).

3 Problem resolution

3.1 Datasets

In this issue, I tried to work initially with several databases provided by BPLim, which consist of product datasets from different supermarkets, where there is information such as the serial number, name, brand, capacity and units of each product. Unfortunately, it has not been possible to advance with the tests using the five datasets provided by BPLim due to the lack of ground truth, which is essential for validating true positives, false negatives, false positives, and true negatives, making reliable evaluation impossible. Additionally, the datasets do not share a common field across all of them, hindering the integration and comparison of data necessary for comprehensive analysis.

So I decided to opt for a database available on Kaggle that addresses the same concept of different product listings, which in this case are products available on Ecommerce sites in Turkey.

In this study, two datasets, each containing 1000 instances, were utilized to advance the analysis of record linkage. The strategic selection and use of these datasets enabled a robust evaluation of the linkage algorithms, significantly enhancing the reliability and validity of the results. The efficiency of this approach was instrumental in producing substantial and meaningful outcomes, demonstrating the practical applicability and effectiveness of the proposed methods in real-world scenarios.

It is important to note that if the datasets had contained more instances, the project would not have been feasible due to the excessive time required by the algorithm to produce results. Even with the current dataset size, processing the linkages between two instances with 1000 rows already demands considerable time, highlighting the importance of optimizing algorithm efficiency for larger datasets. Additionally, while not comparing multiple datasets was an option to maintain efficiency, the algorithm was designed to handle linkages between more than two datasets. However, for this study, it was tested with just two datasets to manage the computational load effectively.

This case consists of two datasets, both with a thousand products and their descriptions, accompanied by the appropriate ground truth where it is possible to know that, after pre-processing the data, there are three hundred and eighteen true matches.

Table 1 - Examples of true matches between the two tested datasets.

Dataset 1		Dataset 2	
Product	**Description**	**Product**	**Description**
SAMSUNG G850 GALAXY ALPHA WHITE SMARTPHONE	SAMSUG G850 GALAXi ALPHA WHITE SMARTPHONE: Weight (gr.) 120 Memori (MB) 2 GB Ram Bluetooth ies Dimenzions (cm) 132.8 x 65.6 x 6.7mm SCREEN SIZE 4.7" Scren Resolution (Pixel) 1280 x 720 Screen Tipe Super AMOLED GPRS GPS, GLONAsss Waranti Period(iear) 2 Processsor Sped (Mhz) A15 1.8GHz+A7 1.3GHZ Operating Sistem Android Camera Resolution (Pixel) 12 MP Keiboard Tipe Touch BRAND samsng MODEL G850FQ Batteri 1860 mAh Color WHITE Storage Capaciti (Mb) 32 GB USB ies 3G ies Wi- Fi (Wirelesss Conection) 802.11 a/b/g/n/ac GPS GPS, GLONAsss Toch Scren ies Product Origin KOREA	SAMSUNG G850 GALAXY ALPHA WHITE SMARTPHONE	SAMSUNG G850 GALAXY ALPHA WHITE SMARTPHONE: Weight (gr.) 120 Memory (MB) 2 GB Ram Bluetooth Yes Dimensions (cm) 132.8 x 65.6 x 6.7mm SCREEN SIZE 4.7" Screen Resolution (Pixel) 1280 x 720 Screen Type Super AMOLED GPRS GPS, GLONASS Warranty Period(Year) 2 Processor Speed (Mhz) A15 1.8GHz+A7 1.3GHZ Operating System Android Camera Resolution (Pixel) 12 MP Keyboard Type Touch BRAND SAMSUNG MODEL G850FQ Battery 1860 mAh Color WHITE Storage Capacity (Mb) 32 GB USB Yes 3G Yes Wi- Fi (Wireless Connection) 802.11 a/b/g/n/ac GPS GPS, GLONASS Touch Screen Yes Product Origin KOREA
Philips 18.5inch 193V5LSB2/62 5ms Led Monitor	Philips 18.5inch 193V5LSB2/62 5ms Led Monnitor: Panel Tipe LED Screen Size 18.5 inch Respons Time 5 ms Conection Tipe Analog Reslution 1366 x 768 Color Black No Pivot No Height Adjustment Vesa Compatible Available	Philips 18.5inch 193V5LSB2/62 5ms Led Monitor	Philips 18.5inch 193V5LSB2/62 5ms Led Monitor: Panel Type LED Screen Size 18.5 inch Response Time 5 ms Connection Type Analog Resolution 1366 x 768 Color Black No Pivot No Height Adjustment Vesa Compatible Available Warranty

	Waranti Period 24 Moths		Period 24 Months
Samsung Galaxy Alpha G850 White Mobile Phone	samsng Galaxi Alpha G850 White Mobile Phone: WArANTi 2 iears Waranti Camera 12 MP Camera RAM Capaciti 2 GB Screen Size 4.7" Touch Screen Bluetooth Available Phone Dimensions (mm) 132.4 x 65.5 x 6.7 mm COLOR White Screen 4.7" Touch Screen Internal Memori 32 GB Internal Memori Operating Sistem Android OS v4.4.4 (KitKat) Weight 115 g FM Radio No Screen Features 1280 x 720 Sms ies Mms ies Video Plaier ies MP3 ies 3G ies Batteri Tipe 1860 mAh GPRS ies	Samsung Galaxy Alpha G850 White Mobile Phone	Samsung Galaxy Alpha G850 White Mobile Phone: WARRANTY 2 Years Warranty Camera 12 MP Camera RAM Capacity 2 GB Screen Size 4.7" Touch Screen Bluetooth Available Phone Dimensions (mm) 132.4 x 65.5 x 6.7 mm COLOR White Screen 4.7" Touch Screen Internal Memory 32 GB Internal Memory Operating System Android OS v4.4.4 (KitKat) Weight 115 g FM Radio No Screen Features 1280 x 720 Sms Yes Mms Yes Video Player Yes MP3 Yes 3G Yes Battery Type 1860 mAh GPRS Yes
SANDISK SDCZ52-008G-B35 8GB USB 2.0, Switch, Flash Memory	SANDISK SDCZ52-008G-B35 8GB USB 2.0, Switch, Flash Memori: CAPACITi 8 GB	SANDISK SDCZ52-008G-B35 8GB USB 2.0, Switch, Flash Memory	SANDISK SDCZ52-008G-B35 8GB USB 2.0, Switch, Flash Memory: CAPACITY 8 GB
SAMSUNG UE48H6270 SMART FHD 3D DVB-S LED LCD TV + 2 Glasses	samsng UE48H6270 SMART FHD 3D DVB-S LED LCD TV + 2 Glassses: Resolution (pixels) 1920 SMART TV ies 3D ies Wifi ies Sistem Frequenci 200 Hz (CMR) Product Origin Imported	SAMSUNG UE48H6270 SMART FHD 3D DVB-S LED LCD TV + 2 Glasses	SAMSUNG UE48H6270 SMART FHD 3D DVB-S LED LCD TV + 2 Glasses: Resolution (pixels) 1920 SMART TV Yes 3D Yes Wifi Yes System Frequency 200 Hz (CMR) Product Origin Imported
Logitech G230 981-	Logitec G230 981-000540 Gaming Headset:	Logitech G230 981-	Logitech G230 981-000540 Gaming Headset:

000540 Gaming Headset	Available for Gaming Brand Logitec conection Tipe Wired Product Tipe Headset with Microfone Usage Method On-Ear Headband Volume Control ies	000540 Gaming Headset	Available for Gaming Brand Logitech Connection Type Wired Product Type Headset with Microphone Usage Method On-Ear Headband Volume Control Yes
Apple iPod Shuffle 2GB Purple MP3/MP4 Player (MD777TZ/A)	Apple iPod Shufle 2GB Purple MP3/MP4 Plaier (MD777TZ/A): COLOR Purple MEMORi 2 GB	Apple iPod Shuffle 2GB Purple MP3/MP4 Player (MD777TZ/A)	Apple iPod Shuffle 2GB Purple MP3/MP4 Player (MD777TZ/A): COLOR Purple MEMORY 2 GB
KINGSTON HyperX Savage 4GB DDR3-1600MHz Memory - HX316C9SR/4	KINGSTON HiperX Savage 4GB DDR3-1600MHz Memori - HX316C9SR/4: Memori Memori Brand Kingston Price Range 100 - 199 TL Platform Desktop Computer Tipe DDR3 Capaciti 4 GB Speed 1600 MHz	KINGSTON HyperX Savage 4GB DDR3-1600MHz Memory - HX316C9SR/4	KINGSTON HyperX Savage 4GB DDR3-1600MHz Memory - HX316C9SR/4: Memory Memory Brand Kingston Price Range 100 - 199 TL Platform Desktop Computer Type DDR3 Capacity 4 GB Speed 1600 MHz

3.1.1 Pre-processing

The preprocessing steps involved removing null values from the "Description" field, ensuring that the "Description" fields from both datasets were in text format, tokenizing the strings for calculating cosine similarity, eliminating duplicate products from both datasets, and trimming the values in the "Description" field. This resulted in dataset one containing five hundred fifty-eight products and dataset two containing five hundred seventy products, with three hundred eighteen true matches between them. No errors were corrected, as the purpose of the project was to develop an algorithm capable of finding matches despite the presence of typos.

3.2 Proposed resolution

My proposal to solve this problem is to develop an algorithm in R that can provide most efficient answer to the presented context, taking into account various record linkage methodologies. The best parameterization was defined by the best results obtained from the set experimentation in the evaluation process.

A meticulous consideration of various similarity metrics was imperative to ensure a comprehensive and robust approach. After careful deliberation, it was determined that a combination of Jaro-Winkler Similarity, Levenshtein Distance, Smith-Waterman Similarity, Q-gram Similarity, and Cosine Similarity would best suit the task at hand.

Each of these similarity metrics possesses unique characteristics and excels in capturing specific aspects of string similarity. The Jaro-Winkler Similarity, for instance, is renowned for its effectiveness in identifying similarities between strings despite variations in spelling and minor typographical errors. Its emphasis on the first few characters of strings enhances its capability to detect similarities even in cases of partial matches.

On the other hand, Levenshtein Distance provides a measure of the minimum number of single-character edits required to transform one string into another. This metric is particularly adept at capturing the degree of dissimilarity between strings, making it invaluable in scenarios where precise matches are elusive.

Smith-Waterman Similarity, with its focus on local alignments and the identification of similar substrings within strings, offers unparalleled sensitivity to subtle similarities

obscured by noise or variations. Its ability to detect similarities at the sub-string level enhances the algorithm's capacity to identify matches in datasets with complex structures.

Q-gram Similarity, by breaking strings into smaller n-gram components and quantifying the similarity based on shared q-grams, provides a robust measure of similarity that is resilient to variations in string length or order. This metric is particularly effective in capturing similarities between strings with different word arrangements or variations in word order.

Finally, Cosine Similarity, rooted in vector space modelling, offers a holistic approach to measuring similarity by considering the frequency of terms in the strings. Its ability to capture semantic similarity and structural resemblance between strings makes it a versatile metric in diverse record linkage scenarios.

By integrating these diverse similarity metrics into the algorithm, I aimed to harness the unique strengths of each metric while mitigating their limitations. The synergy achieved through their combination facilitates a more nuanced and comprehensive assessment of string similarity, enhancing the algorithm's efficacy in record linkage tasks. Indeed, each similarity metric contributes a distinct perspective to the overall analysis, culminating in a holistic approach that excels in capturing the intricacies of string similarity.

First of all I went through to integrate the methods applied as functions as you can see in the image below:

```
# Function to calculate Jaro-Winkler similarity
jaro_winkler_similarity <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "jw"))
}

# Function to calculate Levenshtein distance
levenshtein_distance <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "lv"))
}

# Function to calculate Smith-Waterman similarity
smith_waterman_similarity <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "osa"))
}
```

```
# Function to calculate Q-gram similarity
qgram_similarity <- function(string1, string2, q) {
  return(stringdist::stringdistmatrix(string1, string2, method = "qgram", q = q)[1,1])
}

# Function to calculate Cosine similarity with equal weight and normalization
cosine_similarity_equal_weight <- function(string1, string2) {

# Tokenize strings
  tokens1 <- unlist(strsplit(string1, "\\s+"))
  tokens2 <- unlist(strsplit(string2, "\\s+"))
```

Figure 5 Integration of similarity methods

In this initial process of integrating similarity functions there were no major difficulties since most of the methods are in the stringdist R pacakge and it was enough to construct a simple formula to design its functions. However, Cosine similarity required special attention and went through six steps in the design in its function which were as follows:

1. Tokenization: The function splits the input strings into tokens (words) using whitespace as the delimiter.

```
# Tokenize strings
  tokens1 <- unlist(strsplit(string1, "\\s+"))
  tokens2 <- unlist(strsplit(string2, "\\s+"))
```

Figure 6 Tokenization

2. Vocabulary creation: creation of a combined vocabulary by merging the unique tokens from both strings.

```
# Create a combined vocabulary
  vocabulary <- unique(c(tokens1, tokens2))
```

Figure 7 Vocabulary creation

3. Term Frequency Vectors: For each string, it generates a term frequency vector, where each element represents the frequency of a token in the vocabulary.

```
# Create term frequency vectors
tf1 <- table(tokens1)
tf2 <- table(tokens2)
```

Figure 8 Term Frequency Vectors

4. Numeric Vectors: Convertion of frequency vectors into numeric vectors, maintaining the order of tokens based on the vocabulary.

```
# Convert term frequency vectors to numeric vectors
tf_vector1 <- as.numeric(tf1[match(vocabulary, names(tf1))])
tf_vector2 <- as.numeric(tf2[match(vocabulary, names(tf2))])
```

Figure 9 Numeric Vectors

5. Cosine Similarity Calculation: The cosine similarity is computed as the dot product of the two numeric vectors divided by the product of their Euclidean magnitudes.

```
# Compute cosine similarity
cosine_similarity <- sum(tf_vector1 * tf_vector2) / (sqrt(sum(tf_vector1^2)) *
sqrt(sum(tf_vector2^2)))
```

Figure 10 Cosine Similarity Calculation

Having said that, a metric that was able to take into account all the functions previously introduced began to be designed and it was detected that these metrics needed to have the same weight in the evaluation of the final metric. Then all functions were normalized.

```
# Normalize Jaro-Winkler similarity to range [0,1]
jw_similarity <- jw_similarity/max(jw_similarity,1)
total_distance <- total_distance + jw_similarity

# Normalize cosine similarity to range [0,1]
normalized_cosine_similarity <-cosine_similarity / max(cosine_similarity,1)

# Normalize Smith-Waterman similarity to range [0,1]
sw_similarity <- sw_similarity / max(sw_similarity, 1)
total distance <- total distance + sw_similarity
```

```
# Normalize Q-gram similarity to range [0,1]
qgram_sim <- qgram_sim / max (qgram_sim, 1)
total_distance <- total_distance + qgram_sim
```

Figure 11 Normalization of the functions

Since Levenshtein distance could present negative values, it was necessary to make a previous adaptation in which we calculated the absolute values of the distance before its normalization.

```
# Take the absolute value of Levenshtein distance before normalization
lv_distance <- abs(lv_distance)

# Normalize Levenshtein distance to range [0, 1]
lv_distance <- lv_distance / max(lv_distance, 1)
total_distance <- total_distance + lv_distance
```

Figure 12 Calculation of absolute and normalization of Levenshtein values

The final design of the algorithm consists of record linkage between two datasets using the similarity metrics described above. For each pair of records (one from each dataset), it calculates a total distance score by summing up the individual similarity scores computed using different metrics. If the total distance is below a specified threshold, the pair of records is considered a potential match.

```
# Loop through records and compare similarity scores
  for (i in 1:nrow(dataset1)) {
    for (j in 1:nrow(dataset2)) {
      total_distance <- 0

# Calculate Jaro-Winkler similarity
jw_similarity <- jaro_winkler_similarity(as.character(dataset1[i, "Description"]), as.character(dataset2[j, "Description"]))

# Check for missing values and division by zero
      if (!is.na(jw_similarity) && jw_similarity != Inf) {
        # Normalize Jaro-Winkler similarity to range [0, 1]
```

```
      jw_similarity <- jw_similarity / max(jw_similarity, 1)
      total_distance <- total_distance + jw_similarity
    }

# Calculate Levenshtein distance
    lv_distance <- levenshtein_distance(as.character(dataset1[i, "Description"]), as.character(dataset2[j, "Description"]))


# Check for missing values and division by zero
    if (!is.na(lv_distance) && lv_distance != Inf) {

# Take the absolute value of Levenshtein distance before normalization
      lv_distance <- abs(lv_distance)
      # Normalize Levenshtein distance to range [0, 1]
      lv_distance <- lv_distance / max(lv_distance, 1)
      total_distance <- total_distance + lv_distance
    }

# Calculate Smith-Waterman similarity
  sw_similarity <- smith_waterman_similarity(as.character(dataset1[i, "Description"]), as.character(dataset2[j, "Description"]))

# Check for missing values and division by zero
    if (!is.na(sw_similarity) && sw_similarity != Inf) {
# Normalize Smith-Waterman similarity to range [0, 1]
      sw_similarity <- sw_similarity / max(sw_similarity, 1)
      total_distance <- total_distance + sw_similarity
    }

# Calculate Q-gram similarity
    qgram_sim <- qgram_similarity(as.character(dataset1[i, "Description"]), as.character(dataset2[j, "Description"]), q)


# Check for missing values and division by zero
    if (!is.na(qgram_sim) && qgram_sim != Inf) {
```

```
      # Normalize Q-gram similarity to range [0, 1]
      qgram_sim <- qgram_sim / max(qgram_sim, 1)
      total_distance <- total_distance + qgram_sim
    }


# Calculate Cosine similarity with equal weight
    cosine_sim <- cosine_similarity_equal_weight(as.character(dataset1[i, "Description"]),
as.character(dataset2[j, "Description"]))


# Check for missing values and division by zero
    if (!is.na(cosine_sim) && cosine_sim != Inf) {
      total_distance <- total_distance + cosine_sim
    }

# Check if total distance is below the threshold
    if (!is.na(total_distance) && total_distance <= threshold) {
      link <- cbind(dataset1[i,], dataset2[j,], total_distance)
      links <- rbind(links, link)
    }
  }
 }

 return(links)
}
```

Figure 13 Finalized algorithm

Finally, this algorithm has two adjustable parameters, which were taken into account for the final process of tuning the algorithm and successive tests and analysis of results that are the parameter "Threshold" which consists of a maximum value that we assign to the maximum distance that we want to allow in the sum total of the calculated distances, and the parametron "q", which consists of length of the substrings analyzed in the metric of Q-grams.

```
#Example usage

threshold <-2.2
q <- 700

dataset1 <- read_excel("C:/Users/bruno/OneDrive/Ambiente de Trabalho/dataset 1/Raw_Entire_Dataset 1.2.xlsx")
dataset2 <- read_excel("C:/Users/bruno/OneDrive/Ambiente de Trabalho/dataset 2/Raw_Entire_Dataset 2.2.xlsx")

links<- record_linkage_equal_weight_threshold (dataset1, dataset2, threshold,q)
```

Figure 14 Parametrization and execution of the function

3.3 Experimentation set evaluation

This stage, as the title indicates, is the phase in which an evaluation of the different methodologies applied in this project will be carried out in order to define the best method to solve the main problem of this thesis. This process will consist of using three metrics: Recall, Precision and F-score.

Recall - Defined as $rec = \frac{TP}{TP+FN}$ and it is also known as sensitivity, it is the proportion of actual matches that have been classified correctly. Sensitivity is a common measure in epidemiological studies.

Precision - Measured as $prec = \frac{TP}{TP+FP}$ and is also called positive predictor value. It is the proportion of classified matches that are true matches.

F-score - Is the harmonic mean of precision and recall and is calculated as $f - score = 2 * (\frac{prec*rec}{prec+rec})$. It will have a high value only when both precision and recall have high values and can be seen as a way to find the best compromise between precision and recall.

3.3.1 Experimentation

After some experiments, the first relevant results emerged when the threshold parameter was defined with the value of "2.1" and q with the value of "1000", in which I had 133 Matches, being 129 true positives, 4 false positives and given that it had the

ground truth it was possible to know that they were left to link 189 matches, which means 189 false negatives. With this information it was possible to evaluate, with the previously established evaluation metrics, that this tuning of the algorithm presented 0.4057 recall, 0.9699 Precision and an F-score of 0.5721. In overall, these results indicate that the model has a high precision but a low recall, meaning that it performs well in identifying true positives but misses a significant number of positive instances.

Table 2 - First experiment results.

Links_2.1_1000			
FP	4	**Recall**	0.40566
TP	129	**Precision**	0.969925
FN	189		
		F-score	0.572062

In order to increase the recall and stabilize the precision value, I decided to keep the threshold parameter value of "2.1" and adjusted it to the value of q for "600". In this scenario I obtained 132 Matches, of which 128 are true positives, 4 false positives and 190 false negatives. The recall calculated was 0.4025, while the Precision was 0.9697 and the F-score was 0.5689. In this case, the conclusion was practically the same as in the previous experience: the model performs well identifying true positives but misses a significant number of positive instances.

Table 3 - Second experiment results.

Links_2.1_600			
FP	4	**Recall**	0.402516
TP	128	**Precision**	0.969697
FN	190		
		F-score	0.568889

Keeping the objective indicated in the previous paragraph, I decided to adjust the threshold parameter to "2.2" and the parameter of q to "500", eventually obtaining 266 Matches, of which 246 were true positives, 20 false positives and 72 false negatives. In this case the recall was 0.7736, the Precision was 0.9248 and the F-score was 0.8425.

Table 4 - Third experiment results.

Links_2.2_500			
FP	20	**Recall**	0.773585
TP	246	**Precision**	0.924812
FN	72		
		F-score	0.842466

With the perception of a large margin of improvement in the recall, I decided to keep the threshold parameter to "2.2" and change the parameter from q to "600", eventually obtaining 286 Matches, of which 264 were true positives, 22 false positives and 54 false negatives. In this case the recall was 0.8302, the Precision was 0.9231 and the F-score was 0.8742.

Table 5 - Fourth experiment results.

Links_2.2_600			
FP	22	**Recall**	0.830189
TP	264	**Precision**	0.923077
FN	54		
		F-score	0.874172

Keeping the objective indicated in the previous paragraph, I decided to keep the threshold parameter to "2.2" and adjust the parameter of q to "700", eventually obtaining 298 Matches, of which 275 were true positives, 23 false positives and 43 false negatives. In this case the recall was 0.8648, the Precision was 0.9228 and the F-score was 0.8929.

Table 6 - Fifth experiment results

Links_2.2_700			
FP	23	**Recall**	0.86478
TP	275	**Precision**	0.922819
FN	43		
		F-score	0.892857

3.3.2 Analysis of results conclusion

After this analysis of results it was possible to conclude that tuning algorithm with the threshold parameter with the value "2.2" and with parameter q with the value "700" proves to be the most appropriate for the context initially presented, given that although it that isn't the precision the highest obtained in the experimentation process, it has a high precision value of 0.9228 and the highest recall obtained with a value of 0.8648 and the highest F-score obtained with the value of 0.8929, which allows us to conclude that this algorithm with this tuning demonstrates, that it performs well in identifying true positives while also capturing a significant portion of all actual positive instances, is well-suited for the task at hand and can make reliable predictions with confidence.

4 Conclusion

This thesis delved into the topic of record linkage, a critical aspect of data activities across various sectors, from industry to healthcare. The focus was placed on developing a robust data mining process to interconnect data from unmatched sources. This process facilitates record linkage and consists of meticulous measurements to decipher the matching of data entries.

This work began with an exploration existing bibliography on Record Linkage and similar case studies to the problem at hand. This allowed the description of the process and the distinction between deterministic and probabilistic linkage methods. Various evaluation methods were employed in different character analysis contexts, examining in detail the methodologies of Q-gram, Cosine, Jaro, Levenshtein, Jaro-Winkler, and Smith-Waterman.

Following this bibliography review, the problem statement was introduced, explaining the datasets employed, the proposed solution methodology, and its evaluation, presenting selected metrics with justifications.

Through rigorous investigation and algorithm development in R, incorporating the methodologies explored, the most efficient algorithm and optimal parameterization for solving the case at hand were developed. The experimentation led us to fine-tune the algorithm, setting the threshold parameter to "2.2" and the q parameter to "700." While this configuration did not yield the highest precision, it achieved a commendable precision of 0.9228, coupled with the highest recall of 0.8648 and the highest F-score of 0.8929. This proves that the algorithm is efficient in record linkage tasks, validating its suitability for the presented context.

In conclusion, this thesis has contributed to the advancement of record linkage methodologies, offering insights into algorithmic approaches and parameter optimizations that can enhance the efficiency of data mining processes in diverse domains. The developed algorithm can be applied to develop innovative solutions that address complex data challenges in multiple sectors.

5 References

Bid. , J. M. (1981). Identification of Common Molecular Subsequences . pp. 195-197 .

Boyd, D., & Crawford, K. (2012). Critical Questions for Big Data: Provocations for a Cultural, Technological, and Scholarly Phenomenon. *Information, Communication & Society*, 662-679.

Chetty, R., Friedman, J. N., & Rockoff, J. E. (2014). Measuring the impacts of teachers II: Teacher value-added and student outcomes in adulthood. *American Economic Review*, 2633-2679.

Chickering, D. M., Heckerman, D., & Meek, C. (2004). Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research*, 1287-1330.

Christen, P. (2006). A Comparison of Personal Name Matching: Techniques and Practical Issues. *Sixth IEEE International Conference on Data Mining - Workshops (ICDMW06)*, pp. pp. 290–294.

Christen, P. (2012). *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection.* Springer.

Christen, P., & Goiser, K. (2007). Quality and Complexity Measures for Data Linkage and Deduplication. *ality Measures in Data Mining Studies in Computational Intelligence*, pp. 127–151.

Dean, J., & Barroso, L. A. (2013). The Tail at Scale. *Communications of the ACM*, 74-80.

Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, 107-113.

Denman, K., & Fortier-Shultz, V. (2019, July). Assessing Record Linkage Matches Using String Distance Measures. *New Mexico Statistical Analysis Center*, pp. 1-26.

Dunn, H. L. (1946). "Record Linkage.". *American Journal of Public Health and the Nation's Health*, 1412-1416.

Dwork, C. (2008). Differential Privacy: A Survey of Results. *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation*, 1-19.

Ebraheem, M. (2018). Distributed Representations of Tuples for Entity Resolution. *Proceedings of the VLDB Endowment*, 1454-1467.

Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007, January). Duplicate Record Detection: A Survey. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, vol.19, nº 1*, pp. 1-16.

Fawcett, T. (2006). An Introduction to ROC Analysis. *Pattern Recognition Letters*, 861-874.

Fellegi, I. P., & Sunter, A. B. (1969). A Theory for Record Linkage, 64(328), 1183-1210. *Journal of the American Statistical Association*, 1183-1210.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian Network Classifiers. *Machine Learning*, 131-163.

Goeken, R. (2011). New methods of census record linking. *Historical Methods: A Journal of Quantitative and Interdisciplinary History*, 7-14.

Gözükara, F. (2021, 09 03). An Incremental Hierarchical Clustering Based System For Record Linkage In E-Commerce Domain. *The Computer Journal*, 581-602. Retrieved from Kaggle.

Hassanzadeh, O., & Miller, R. J. (2009). "A Framework for Uncertainty in Data Integration.". *Proceedings of the VLDB Endowment*, 1381-1392.

Hernandez, M. A., & Stolfo, S. J. (1995). The Merge/Purge Problem for Large Databases. *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, (pp. 127-138).

Homlund, O. (2019). EVALUATING RECORD LINKAGE METHODS FOR MANIFOLD IDENTITY DETECTION (Master's thesis). *Umeå University, Sweden*, pp. 1-50.

Indyk, P., & Motwani, R. (1998). Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, 604-613.

Jensen, P. B., Jensen, L. J., & Brunak, S. (2012). Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 395-405.

Karau, H. (2015). Learning Spark: Lightning-Fast Big Data Analysis. *O'Reilly Media*.

Kling, J. R. (2006). Incarceration length, employment, and earnings. *American Economic Review*, 863-876.

Koudas, N., Sarawagi, S., & Srivastava, D. (2006). Record Linkage: Similarity Measures and Algorithms. *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, 802-803.

Larmuseau, M. H. (2014). The impact of recent connections on the genetic legacy of past migrations: A case study of Belgian genealogical records. *American Journal of Physical Anthropology*, 540-547.

Leskovec, J., Rajaraman, A., & Ullman, J. D. (2020). Mining of Massive Datasets. *Cambridge University Press*.

Little, R. J., & Rubin, D. B. (2019). Statistical Analysis with Missing Data. *John Wiley & Sons*.

Manning, C. D., Raghavan, P., & & Schütze, H. (2008). Introduction to Information Retrieval . *Cambridge University Press*.

Mudgal, S. (2018). Deep Learning for Entity Matching: A Design Space Exploration. *Proceedings of the 2018 International Conference on Management of Data (SIGMOD '18)*, 19-34.

Nakamura, E. (2017). Record Linkage in the Blockchain Era: Linking Data Without Sharing the Underlying Information. *Proceedings of the 2017 IEEE International Conference on Big Data*, 4456-4459.

Narayanan, A., & Shmatikov, V. (2008). Robust De-anonymization of Large Sparse Datasets. *IEEE Symposium on Security and Privacy*.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys, vol. 33, nº1*, pp. 31-88.

Newcombe, H. B., Kennedy, J. M., Axford, S. J., & James, A. P. (1959). 2. (). "Automatic Linkage of Vital Records." Science, 130(3381), . *Science*, 954-959.

Rahm, E., & Do, H. H. (2000). Data Cleaning: Problems and Current Approaches. *IEEE Data Engineering Bulletin*, 3-13.

Reardon, S. F. (2017). Educational inequality, race, and parental income: The roots of the widening achievement gap. *Russell Sage Foundation Journal of the Social Sciences*, 1-42.

Ruggles, S. (2006). Linking historical censuses: A new approach. *History and Computing*, 213-224.

Ruggles, S. (2011). The North Atlantic Population Project: Progress and Prospects. *Historical Methods: A Journal of Quantitative and Interdisciplinary History*, 1-6.

Schafer, J. L., & Graham, J. W. (2002). Missing Data: Our View of the State of the Art. *Psychological Methods*, 147-177.

Stevens, D. J. (2006). Linking criminal history records: A research perspective. *Journal of Economic and Social Measurement*, 159-175.

Sweeney, L. (2002). "k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 557-570.

Talburt, J. R. (2011). Entity Resolution and Information Quality. *Morgan Kaufmann.*

Union, E. (2016). General Data Protection Regulation (GDPR). *Official Journal of the European Union.*

Westra, B. L. (2011). Big data and knowledge synthesis in nursing: Implications for health policy. *Nursing Outlook*, 4-10.

White, T. (2015). Hadoop: The Definitive Guide. *O'Reilly Media.*

Wilson, D. R. (2011, August 5). Beyond Probabilistic Record Linkage: Using Neural Networks and Complex Features to Improve Genealogical Record Linkage. *Proceedings of International Joint Conference on Neural Networks*, pp. 9-14.

Winkler , W. E., & Thibaudeau, Y. (1991). AN APPLICATION OF THE FELLEGI-SUNTER MODEL OF RECORD LINKAGE TO THE 1990 U.S. DECENNIAL CENSUS. *Technical Report Statistical Research Report Series RR91/09, US Bureau of the Census, Washington D.C.*

Winkler, W. E. (1994). Advanced Methods for Record Linkage. *Proceedings of the Section on Survey Research Methods, American Statistical Association*, 467-472.

Winkler, W. E. (1999). *The State of Record Linkage and Current Research Problems Statistical Research Division, U.S. Census Bureau.* Statistical Research Division, U.S. Census Bureau.

Winkler, W. E. (2002). Methods for Record Linkage and Bayesian Networks . *Statistical Research Division, U.S. Census Bureau.*

Zaharia, M. (2010). Spark: Cluster Computing with Working Sets. *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 10-10.

Zook, M. (2017). Ten simple rules for responsible big data research. *PLOS Computational Biology.*

Zyskind, G., Nathan, O., & Pentland, A. (2015). Decentralizing Privacy: Using Blockchain to Protect Personal Data. *IEEE Security and Privacy Workshops*, 180-184.

6 Attachments

6.1 Gantt's Diagram

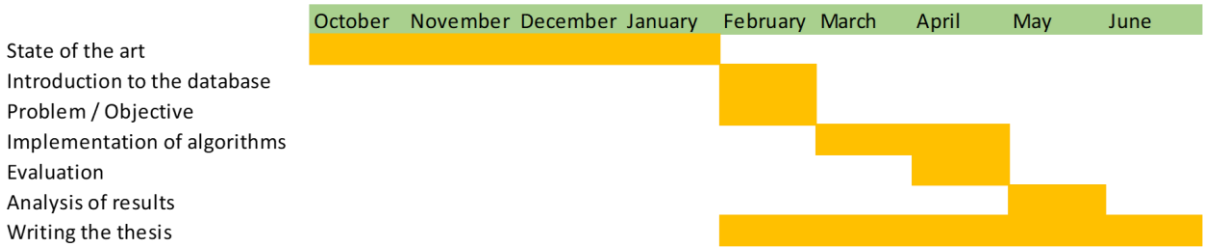


Figure 15 Gantt's diagram

6.2 Algortihm Code

```
library(stringdist)
library(readxl)

# Function to calculate Jaro-Winkler similarity
jaro_winkler_similarity <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "jw"))
}

# Function to calculate Levenshtein distance
levenshtein_distance <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "lv"))
}

# Function to calculate Smith-Waterman similarity
smith_waterman_similarity <- function(string1, string2) {
  return(stringdist::stringdist(string1, string2, method = "osa"))
}

# Function to calculate Q-gram similarity
qgram_similarity <- function(string1, string2, q) {
  return(stringdist::stringdistmatrix(string1, string2, method =
"qgram", q = q)[1,1])
}

# Function to calculate Cosine similarity with equal weight and
normalization
cosine_similarity_equal_weight <- function(string1, string2) {
  # Tokenize strings
  tokens1 <- unlist(strsplit(string1, "\\s+"))
  tokens2 <- unlist(strsplit(string2, "\\s+"))

  # Create a combined vocabulary
  vocabulary <- unique(c(tokens1, tokens2))

  # Create term frequency vectors
  tf1 <- table(tokens1)
  tf2 <- table(tokens2)
```

```
  # Convert term frequency vectors to numeric vectors
  tf_vector1 <- as.numeric(tf1[match(vocabulary, names(tf1))])
  tf_vector2 <- as.numeric(tf2[match(vocabulary, names(tf2))])

  # Compute cosine similarity
  cosine_similarity <- sum(tf_vector1 * tf_vector2) /
(sqrt(sum(tf_vector1^2)) * sqrt(sum(tf_vector2^2)))

  # Normalize cosine similarity to range [0, 1]
  normalized_cosine_similarity <- cosine_similarity /
max(cosine_similarity, 1)

  return(normalized_cosine_similarity)
}

# Function to perform record linkage using multiple similarity
metrics with equal weight and a single threshold
record_linkage_equal_weight_threshold <- function(dataset1,
dataset2, threshold, q) {
  links <- data.frame()

  # Loop through records and compare similarity scores
  for (i in 1:nrow(dataset1)) {
    for (j in 1:nrow(dataset2)) {
      total_distance <- 0

      # Calculate Jaro-Winkler similarity
      jw_similarity <-
jaro_winkler_similarity(as.character(dataset1[i, "Description"]),
as.character(dataset2[j, "Description"]))

      # Check for missing values and division by zero
      if (!is.na(jw_similarity) && jw_similarity != Inf) {
        # Normalize Jaro-Winkler similarity to range [0, 1]
        jw_similarity <- jw_similarity / max(jw_similarity, 1)
        total_distance <- total_distance + jw_similarity
      }

      # Calculate Levenshtein distance
      lv_distance <- levenshtein_distance(as.character(dataset1[i,
"Description"]), as.character(dataset2[j, "Description"]))

      # Check for missing values and division by zero
      if (!is.na(lv_distance) && lv_distance != Inf) {
        # Take the absolute value of Levenshtein distance before
normalization
        lv_distance <- abs(lv_distance)
        # Normalize Levenshtein distance to range [0, 1]
        lv_distance <- lv_distance / max(lv_distance, 1)
        total_distance <- total_distance + lv_distance
      }

      # Calculate Smith-Waterman similarity
```

```
      sw_similarity <-
smith_waterman_similarity(as.character(dataset1[i, "Description"]),
as.character(dataset2[j, "Description"]))

      # Check for missing values and division by zero
      if (!is.na(sw_similarity) && sw_similarity != Inf) {
        # Normalize Smith-Waterman similarity to range [0, 1]
        sw_similarity <- sw_similarity / max(sw_similarity, 1)
        total_distance <- total_distance + sw_similarity
      }

      # Calculate Q-gram similarity
      qgram_sim <- qgram_similarity(as.character(dataset1[i,
"Description"]), as.character(dataset2[j, "Description"]), q)

      # Check for missing values and division by zero
      if (!is.na(qgram_sim) && qgram_sim != Inf) {
        # Normalize Q-gram similarity to range [0, 1]
        qgram_sim <- qgram_sim / max(qgram_sim, 1)
        total_distance <- total_distance + qgram_sim
      }

      # Calculate Cosine similarity with equal weight
      cosine_sim <-
cosine_similarity_equal_weight(as.character(dataset1[i,
"Description"]), as.character(dataset2[j, "Description"]))

      # Check for missing values and division by zero
      if (!is.na(cosine_sim) && cosine_sim != Inf) {
        total_distance <- total_distance + cosine_sim
      }

      # Check if total distance is below the threshold
      if (!is.na(total_distance) && total_distance <= threshold) {
        link <- cbind(dataset1[i,], dataset2[j,], total_distance)
        links <- rbind(links, link)
      }
    }
  }

  return(links)
}

threshold <- 2.2
q <- 700
links <- record_linkage_equal_weight_threshold(dataset1, dataset2,
threshold, q)
print(links)
```